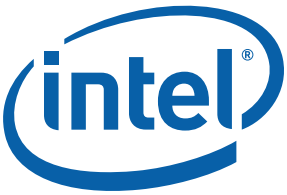


Intel® EP80579 Integrated Processor Product Line

Datasheet

Order Number: 320066-003US
August 2009

Product Features

- System on a Chip (SoC)
  - — Integrated Intel® Architecture (IA) processor and chipset (MCH/ICH) technology
  - — Extensive integration of standard Intel architecture communications interfaces provide cost, power and board area savings (Gigabit Ethernet (GbE), Time Division Multiplexing (TDM)‡ processing, Security Services Unit (SSU),‡ and Acceleration Services Units (ASU)‡)
- SKU Support[1]
  - — Embedded: Intel architecture compatibility and high-speed interfaces (GbEs, PCI Express*)
  - — Application Services: Security — Packet security compatibility and IP Telephony packet security, TDM, and High-Level Data Link Control (HDLC)
- Intel Architecture Processor
  - — Low-power and high-performance architecture based on Intel Architecture (IA-32) processor
  - — Three operating frequency SKUs:
    - 600 MHz, 1066 MHz, or 1200 MHz
  - — 256 KB L2 data coherent cache (2 way)
- Integrated Memory Control Hub (IMCH) and Integrated I/O Control Hub (IICH) Compatible
  - — Enhanced DMA (EDMA) controller
  - — Two SATA Gen1 or Gen2 interfaces
  - — Two USB 1.1 or USB 2.0 ports
  - — Two integrated, 16550-compatible UARTs
  - — LPC 1.1 interface
  - — Serial Peripheral Interface (SPI)†
  - — Two SMBus 2.0 compliant interfaces
  - — GPIOs
  - — Watchdog Timer
  - — One 32/64-bit and two 32-bit high-precision event timers
- Acceleration Services Unit (ASU)‡
  - — High performance accelerator on-chip engines for packet processing
  - — Support capabilities for commonly used protocol implementations such as TCP/IP, UDP, IPSec, SSL, NAT, and SRTP
- Security Services Unit (SSU)‡
  - — High-performance on-chip Crypto Accelerator
  - — Support capabilities for commonly used cryptographic protocol implementations
- Single-Channel Double-Data-Rate (DDR) SDRAM Memory
  - — Supports DDR2 at 400/533/667/800 MT/s
  - — Supports 32 or 64-bit interfaces
  - — Error correction code (ECC); single-bit correct/ double-bit detect (SEC/DED) coverage
  - — Addressable from Intel architecture processor and PCI Express
- Three Gigabit Ethernet MACs
  - — Three 10/100/1000 ports with RGMII/RMII interfaces
  - — MDIO interface for external PHY configuration
  - — Serial EEPROM interface supports network boot and wake-on LAN
- Industry Standard PCI Express Interface
  - — Supports 1x8, 2x4, or 2x1 configurations as a root complex
- Integrated Serial ATA (SATA) Host Controllers
  - — Independent DMA operation on two ports
  - — Data transfer rates up to 3.0 Gb/s
  - — Alternate Device ID
- Integrated High-speed Serial Interface (TDM)‡
  - — Supports up to 12 external T1/E1 and codecs
  - — Supports up to 128 HDLC channels
- Local Expansion Bus (LEB)
  - — Supports up to eight chip selects
  - — 25-bit address and 16-bit data
  - — Supports HPI-8 and HPI-16
- Dual Controller Area Network (CAN)
  - — Supports two CAN 2.0b interfaces
- Single Synchronous Serial Port (SSP) Compatible
- IEEE 1588-2008 Hardware Assistance
  - — Supports two GbE and two CAN interfaces
  - — Time master/target support
- 1088-Ball FCBGA package
  - — Dimensions of 37.5 mm x 37.5 mm
  - — 1.092-mm solder ball pitch
  - — Lead-free only — RoHS 5/6 compliant
- Typical Applications
  - — Embedded, Security and/or IP Telephony applications

† Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.

‡ Feature must be enabled with EP80579 software. Refer to the EP80579 software documentation for more information.

1. For complete information about product features and SKUs, please refer to Chapter 47.0, "SKUs, Power Savings and Pre-Boot Firmware".

Contents

Acceleration and I/O Complex, Volume 4 of 6 1213

Figures

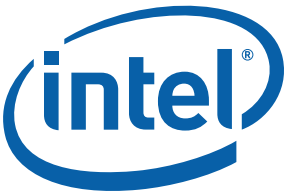
Tables

Revision History

Date	Revision	Description
August 2009	003	Changed the following signal names: • EX_REQ_GNT# to Reserved 19 • EX_SLAVE_CS# to Reserved 20 • EX_GNT_REQ# to NC57 • EX_WAIT# to NC58 • EX_WDTXFER to NC59 Corrected signal name: • SIU_CST1 to SIU_CST1# • SIU_CST2 to SIU_CST2# Updated: • Section Product Features • Table 1-4, “Glossary Table” • Table 2-1, “EP80579 External Interface Summary” • Table 5-32, “Summary of Local Expansion Bus Error Conditions” • Table 6-5, “Powergood Reset Timings” • Section 6.3.2.1, “Transitioning Between Power States” • Section 11.4.6, “RCOMP” • Section 16.5.1.64, “Offset 268h: DDRIOMC2 - DDR IO Mode Control Register 2” • Section 23.1.1.5, “PI - Programming Interface Register” • Section 37.5.11.6.3, “Checksum Word Calculation” • Section 42.5, “Register Summary” • Table 42-7, “EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers” • Table 42-9, “EXP_PARITY_STATUS - Expansion Bus Parity Status Register” • Table 42-10, “LEB Performance Calculation - Estimated AIOC Latencies” • Table 48-24, “Expansion Bus Signals” • Table 48-22, “Gigabit Ethernet Interface Signals” with signal name changes • Figure 49-28, “LPC Valid Delay from Rising Clock Edge Diagram” and Figure 49-32, “IICH Clock (CLK14) Timing Diagram” with signal name corrections • Table 49-10, “Power Management DC Input Characteristics” PWRBTN# pin • Table 49-11, “Power Management DC Output Characteristics” PWRBTN# pin • Table 49-36, “SMBus DC Input Characteristics” Intruder# pin • Table 49-37, “SMBus DC Output Characteristics” Intruder# pin • Table 49-106, “IMCH Reset Signals DC Input Characteristics” CLK100 (***Continued Next Page***)

Date	Revision	Description
August 2009	003	Added: • Industrial temperatures to Chapter 49.0, “Electrical Specifications” and Chapter 50.0, “Thermal Specifications and Design Considerations” The following changes were made due to defeaturing LEB Mastering: • Section 2.2, “Signaling Architecture” • Table 2-1, “EP80579 External Interface Summary” - LEB Description • Table 2-4, “Summary of Communication” - removed “LEB Master” • Table 3-7, “Address Space Sizes of AIOC-attached Devices” - removed LEB row • Note E in Table 3-11, “PCI Configuration Header Support for Type 0 Headers in AIOC Devices” • Section 4.2.2, “Other Agents” - fourth bullet removed “master” • Table 5-32, “Summary of Local Expansion Bus Error Conditions” - Notes description • Removed rows in Table 7-67, “Bus M, Device 8, Function 0: Summary of Local Expansion Bus Registers Mapped Through CSRBAR PCI Memory BAR"” containing LEB content • Removed LEB content from Section 36.3, “Local Expansion Bus Interface (LEB)” • Removed content from Section 42.0, “Local Expansion Bus Controller” •“EXP_MST_CONTROL - Expansion Bus Control Register” •“EXP_LOCK0 - Expansion Bus Lock Register” •“Offset D0500010h: CMD_TRNS1_W[0-3] - Command Translation Window Register “ • Removed LEB content from Section 42.1, “Overview” and Section 42.2, “Feature List” • Removed LEB content from Figure 42-1, “Expansion Bus Controller” • Removed LEB content and also “Inbound Transfers” , “Arbitration”, “Expansion Bus Inbound Timing Diagrams” and “External Expansion Bus Timing Diagram” content from Section 42.4, “Theory of Operation” • Removed Table 42-16 and Section 42.5.2.6 Removed: Removed two rows with D30/D31 content from Table 1-4, Section 6.1.2.3.1, “IICH”, Table 28-9, and from PCIRST# description in Table 48-28.

Date	Revision	Description
December 2008	002	Added: • Chapter 28.0, “IA-32 Core Interface” Updated: • Figure 6-1, “Powergood and Reset Interface” • Table 49-12, “Power Sequencing Signal Timings” • Table 16-26, “Offset 9Ch: DEVPRES - Device Present Register” • Table 16-40, “Offset F6h: IMCH_TST2 - IMCH Test Byte 2 Register” • Section 22.1, “Overview” • Text in Section 35.12.1.9, “Offset 14h: MMBAR – Expansion Bus Base Address Register” • Text in Section 42.5.1.2, “EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers” • Figure 42-2, “Chip Select Address Allocation When There Are no 32-MByte Devices Programmed” • Figure 42-4, “Chip Select Address Allocation when a 32 Mbyte device is programmed” • Figure 48-3, “FCBGA Package — Bottom View” • Table 48-24, “Expansion Bus Signals” • Table 48-29, “Reserved Pin List” • Table 48-30, “No Connect Pin List” • Table 49-7, “Maximum Supply Current Embedded SKU” • Table 49-11, “Power Management DC Output Characteristics” • Table 49-36, “SMBus DC Input Characteristics” • Table 49-38, “SMBus DC Clock Specification” • Table 49-48, “SPI DC Output Characteristics” • Table 49-67, “DC Output Characteristics: RMII Mode of Operation” • Table 49-82, “EEPROM Timing Values” • Table 49-84, “TDM DC Output Characteristics” • Table 49-89, “LEB DC Output Characteristics” • Table 49-93, “CAN DC Output Characteristics” • Table 49-95, “SSP DC Output Characteristics” • Table 49-99, “IEEE 1588-2008 Hardware Assist DC Output Characteristics” • Table 49-101, “IICH Miscellaneous Signals DC Output Characteristics” • Table 49-108, “JTAG DC Output Specifications (BPM4_PRDY_OUT)”
July 2008	001	Initial release of this document.



Introduction and Overview, Volume 1 of 6

Order Number: 320066-003US

1.0 Introduction

1.1 Introduction

The Intel® EP80579 Integrated Processor product line is made of up of the Intel® EP80579 Integrated Processor and the Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology. The Intel® EP80579 Integrated Processor product line is an integrated System On a Chip (SoC). The Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology architecture combines Intel Architecture (IA)-based communications processors, a memory hub controller (IMCH), an I/O architecture (IICH), and high speed I/O interfaces (PCI Express*, Gigabit Ethernet). The Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology also features high-performance packet processing and security capabilities. The Intel® EP80579 Integrated Processor product line architecture is designed to provide best-in-class processing performance, stringent power usage, and reasonable cost targets while maintaining IA implementation and providing the required I/O throughput.

The intended audience for this document is architects, hardware/software design engineers or designer who may need specific technical information for the development and programming of the EP80579 integrated processor SoC. This document is also intended for an audience that has a thorough understanding of IA-32 microprocessor, memory controller and I/O architectures as well as a basic understanding of system software architectures (operating system and pre-boot firmware).

1.2 Document Organization

Note: The Intel® EP80579 Integrated Processor product line is referred to as the "EP80579". In cases where the features are specific to a given processor we will use Intel® EP80579 Integrated Processor or Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology.

This document first provides an overview of the Intel® EP80579 Integrated Processor product line architecture. The overview chapter provides a block diagram and defines the 's external and internal interfaces. This is followed by a functional description of the following blocks:

- IA Complex (including the IA-32 core, IMCH and IICH).
- Acceleration and I/O Complex (including ASU, SSU, and high-speed I/O interfaces such as Gigabit Ethernet and TDM).
- Test and Debug information, including JTAG.
- Technical Specifications (SKUs, Packaging, Electrical and Thermal).

1.3 Referenced Documents and Related Websites

Visit the Intel® EP80579 Integrated Processor product line Website for other information:

http://www.intel.com/go/SoC

Table 1-1. Referenced Documents

Document Title	Location
Advanced Configuration and Power Interface (ACPI) Specification	http://www.acpi.info/
Enhanced Host Controller Specification (EHCI and UHCI)	http://www.intel.com/technology/USB/spec.htm/
IEEE 1149.1: IEEE Standard Test Access Port and Boundary-Scan Architecture	http://ieeexplore.ieee.org
IEEE 1588:Precision Clock Synchronization Protocol for Networked Measurement and Control Systems	http://ieeexplore.ieee.org
Intel Architecture Software Developer's Manual, Volumes 1–3	http://developer.intel.com/design/pentium4/manuals/index_new.htm#sdm_vol1
Intel Corporation, Advanced Host Controller Interface Specification for Serial ATA	http://www.intel.com/technology/serialata/ahci.htm
Intel Corporation, Enhanced Host Controller Interface Specification for Universal Serial Bus	http://www.intel.com/technology/usb/ehcispec.htm
Intel Corporation, Low Pin Count (LPC) Interface Specification	http://www.intel.com/design/chipsets/industry/lpc.htm
Intel Corporation, Multiprocessor Specification	http://www.intel.com/design/archives/processors/pro/docs/242016.htm
Intel Corporation, Universal Host Controller Interface (UHCI) Specification	http://www.intel.com/technology/usb/ehcispec.htm
Intel Corporation, Universal Serial Bus (USB) Specification	http://www.intel.com/technology/usb/spec.htm
Intel Corporation, USB2 Debug Device Functional Specification	http://www.intel.com/technology/usb/download/DebugDeviceSpec_R090.pdf
Intel® 82093AA I/O Advanced Programmable Interrupt Controller (I/O APIC)	http://www.intel.com/design/chipsets/specupdt/290710.htm?iid=search&
Intel® EP80579 Integrated Processor product line Platform Design Guide	http://www.intel.com/go/SoC
Intel® EP80579 Integrated Processor product line Specification Update	http://www.intel.com/go/SoC
Intel® EP80579 Integrated Processor product line Thermal/Mechanical Design Guide	http://www.intel.com/go/SoC
JEDEC Specification	http://www.jedec.org/default.cfm
Low Pin Count Specification (LPC)	http://www.intel.com/design/chipsets/industry/lpc.htm
Serial ATA Specification	http://www.serialata.org/specifications.asp
SMBus Specification	http://www.smbus.org/specs/
Universal Host Controller Specification (EHCI and UHCI)	http://www.intel.com/technology/USB/spec.htm/
Universal Serial Bus Specification	http://www.usb.org/developers/docs/ http://www.intel.com/technology/USB/spec.htm/

Table 1-2. Related Websites

Specification or Technology	Website
AC'97 Rev 2.2 Specification	http://developer.intel.com/ial/scalableplatforms/audio/index.htm #97spec/
ACPI and related specifications	http://www.acpi.info/spec.htm
AT Attachment-6 with Packet Interface (ATA/ATAPI-6)	http://T13.org (T13 1410D)
BIOS Boot Specifications	http://www.phoenix.com/en/customer+services/white+papers-specs/
Communication and Network Riser Rev 1.2 Specification	http://developer.intel.com/technology/cnr/download.htm
Front Panel I/O Connectivity Design Guide	http://www.formfactors.org/DeveloperResources.asp
PCI and PCI Express* related specifications	http://www.pcisig.com/specifications
PIRQ Routing Table Information	http://www.microsoft.com/whdc/archive/pciirq.mspx
Power Management Specifications	http://www.microsoft.com/whdc/resources/respec/specs/pmref/default.mspx

1.4 Acronyms

This section describes acronyms that are used throughout this document.

Table 1-3. Acronym Table

Term	Description
ACPI	Advanced Configuration and Power Interface Specification, an industry specification of the common interfaces enabling robust operating system (OS)-directed motherboard device configuration and power management of both devices and entire systems.
AHCI	Advanced Host Controller Interface, an industry specification of the interface between memory and SATA devices.
AIO	IMCH A-unit I/O Mux Leg
AIOC	Acceleration and I/O Complex
AMC	Audio/Modem Codec
ARP	Address resolution protocol
ASF	Alert Specification Format. This is the next generation of "Alert on LAN*" implementation.
ASU	Acceleration Services Unit
BAR	PCI Base Address Register used to define the base and limit of an I/O or memory region assigned to a PCI device.
BER	Bit Error Rate
BGA	Ball Grid Array
CM	Coherent Memory
CMC	Common Mode Choke
CMI	Core (IA-32 core) interface, Memory controller hub, I/O controller hub
CNR	Communications and Networking Riser
CRC	See Cyclic Redundancy Check in Table 1-4.
CSMA/CD	Carrier Sense Multiple Access/Carrier Detect
DDP	Direct Data Placement Protocol
DDR	DDR SDRAM (Double Data Rate Synchronous Dynamic Random Access Memory) is a system memory technology.

Table 1-3. Acronym Table

Term	Description
DED	Double-bit Error Detect
DMA	See Direct Memory Access in Table 1-4.
DW	Double Word. A legacy reference to 32 bits of data on a naturally aligned four-byte boundary (i.e. the least significant two bits of the byte address are b00). This is a legacy term used by PCI and must not be used other than in that context.
ECC	Error Checking and Correction
EDMA	Enhanced DMA
EMI	Electro Magnetic Interference
EMTS	Electrical Mechanical Thermal Specification used for processor specifications.
EOP/EOF	End Of Packet / End Of Frame
ESD	Electrostatic Discharge
EXP	A generic designation for the I/O interconnect technology also known as PCI Express*.
FRU	Field Replaceable Unit
FS	Full-speed. Refers to USB.
FSB	Front Side Bus (a common external interface for IA processors)
FWH	Firmware Hub. A non-volatile memory device used to store the system BIOS/pre-boot firmware.
GbE	Gigabit Ethernet controller
GMII	Gigabit MII
HBA	Host Bus Adapter - necessary when connecting a peripheral to a computer that doesn't have native support for that peripheral's interface.
HCD	Host Controller Device - USB interface for programmers
HECBASE	PCI Express* Enhanced Configuration Base Register
HPET	High Precision Event Time (HPET) - The IA-PC HPET Architecture defines a set of timers that can be used by the operating system. The timers are defined such that the OS may be able to assign specific timers to be used directly by specific applications. Each timer can be configured to generate a separate interrupt.
HSI	High Speed Interface. Refers to USB.
I/O	1. Input/Output. 2. When used as a qualifier to a transaction type, specifies that transaction targets Intel Architecture™ specific I/O space (e.g., I/O read).
IA	Intel Architecture instruction set commonly known as "x86"
IA-CPU	IA-CPU, IA Complex and IA Processor are the same terminology
ICH	I/O Controller Hub, ICH and IICH are interchangeable for entire document
IICH	Integrated I/O Controller Hub, ICH and IICH are interchangeable for entire document
IMCH	Integrated Memory Controller Hub, MCH and IMCH are interchangeable for the entire document
INTx	Legacy PCI interrupt architecture that encodes interrupts on one of four side-band signals (INTA, INTB, INTC, and INTD).
IP	Internet Protocol
ISA	See Industry Standard Architecture in Table 1-4
LEB	Local Expansion Bus, or LE Bus
LML	Latency Measurement Logic
LPC	Low Pin Count
LS	Low-speed. Refers to USB.
LSb	Least Significant Bit

Table 1-3. Acronym Table

Term	Description
LSB	Least Significant Byte
MCH	Memory Controller Hub, MCH and IMCH are interchangeable for the entire document
MII	Media Independent Interface (16 pins per port)
MMIO	Memory Mapped I/O
MMR	Memory Mapped Register
MSb	Most Significant Bit
MSB	Most Significant Byte
MSI	Message-signaled interrupt that encodes interrupts as an in-band 32-bit write transaction.
MTBF	Mean Time Between Failures
NCM	Non Coherent Memory
NIC	Network interface controller
NOS	Network Operating System
NSI	North South Interface. The designation for the proprietary, internal high-speed serial interconnect between the IMCH and the IICH.
OS	Operating System.
OSPM	Operating System directed Power Management
P2P	See Peer-to-Peer in Table 1-4
PB	Packet Buffer
PBM	Packet Buffer Memory
PCI	Peripheral Component Interconnect Local Bus. A 32- or 64-bit bus with multiplexed address and data lines that is primarily intended for use as an interconnect mechanism within a system between processor/memory and peripheral components or add-in cards.
PCM	Pulse Code Modulation
PEC	Packet Error Checking. This is an SMBUS 2.0 feature.
PHY	Physical Layer Device
POC	Power-on-configuration
RASUM	Reliability, Availability, Serviceability, Usability, and Manageability, which are all important characteristics of servers.
RCBA	Root Complex Base Address register at D31:F0:RegF0h. It specifies the physical address of the CMI Configuration Space. Also used in RCBA + offset xxxxh or RCBA + xxxxh (where xxxxh is the offset) to indicate register location in the CMI Configuration Space.
RCRB	Root Complex Register Block, as defined in the *PCI Express* Specification v1.0a*. In the IICH context, it refers to a part of the CMI Configuration Space (see RCBA, above).
RDMA	Remote Direct Memory Access
RFL	Receive FIFO Level
RGMII	Reduced GMII
RMII	Reduced MII (7 pins per port)
RMW	Read-Modify-Write operation
RTC	Real-Time Clock
RTCRESET#	Signal that resets the RTC well (but does not clear the RTC RAM memory contents).
RX	Receive
SATA	Serial Advanced Technology Attachment
SATA*	Serial ATA, an industry specification of the interface for storage controllers and devices.

Table 1-3. Acronym Table

Term	Description
SEC	Single-bit Error Correct
SEC/DED	Single Error Correct/Double Error Detect - A specific data protection algorithm that distributes data and ECC across 144 bits. Enables correction of single bit errors. Allows detection of double bit errors.
SOP/SOF	Start Of Packet / Start Of Frame
SMM	System Management Mode
SPD	Serial Presence Detect
SSU	Security Services Unit
STR	Suspend To Ram
TAP	Test Access Port used for testability and debug of the component.
TX	Transmit
TCO	Total Cost of Ownership
TCP	Transmission Control Protocol
TDM	Time Division Multiplexed
TDR	Time Domain Reflectometry
TFL	Transit FIFO Level
TID	See Transaction Identifier in Table 1-4
USB	Universal Serial Bus
VCMI	IA-32 core, IA-32 Core interface, Memory controller hub, I/O controller hub
VLAN	Virtual Local Area Network
WDT	Watch Dog Timer

1.5 Glossary

This section presents a glossary for this document.

Table 1-4. Glossary Table (Sheet 1 of 5)

Term	Definition
µBGA	Micro Ball Grid Array
AIOC Direct (AD)	AIOC Direct (AD) memory regions are not coherent with IA caches when accessed from AIOC agents. Accesses to these memory regions enter the memory system through the Memory Controller avoiding the IMCH. Memory regions that are not coherent with IA caches need not be accessible to the IA CPU.
Agent	A logical device connected to a bus or shared interconnect that can either initiate accesses or be the target of accesses.
ALT Access Mode	Mode to allow the reading of write-only registers, usually used when saving/restoring register content for power management sleep state implementations.
Anti-Etch	Any plane-split, void or cutout in a V_{CC} or GND plane is referred to as an anti-etch.
Asserted	Signal is set to a level that represents logical true.
Asynchronous	1. An event that causes a change in state with no relationship to a clock signal. 2. When applied to transactions or a stream of transactions, a classification for those that do not require service within a fixed time interval.

Table 1-4. Glossary Table (Sheet 2 of 5)

Term	Definition
Atomic operation	A series of two or more transactions to a device by the same initiator which are guaranteed to complete without intervening accesses by a different master. Most commonly required for a read-modify-write (RMW) operation.
Block Locking	Ability to lock the FWH's blocks to write-protect, read-protect, or open state.
Buffer	1. A random access memory structure. 2. The term I/O buffer is also used to describe a low-level input receiver and output driver combination.
Cx States	Processor power states (Cx states) are processor power consumption and thermal management states within the global working state, G0. • C0: Processor power state - While the processor is in this state, it executes instructions. • C1: Processor power state - This power state has the lowest latency. The hardware latency in this state must be low enough that the operating software does not consider the latency aspect of the state when deciding whether to use it. • C2: Processor power state - This state offers improved power savings over the C1 state. The worst-case hardware latency for this state is provided via the ACPI system firmware and operating software can use this information to determine when the C1 state should be used instead of the C2 state. • C3: Processor power state - This state is not supported. The C3 state offers improved power savings over the C1 and C2 states. The worst-case hardware latency for this state is provided via the ACPI system firmware and the operating software can use this information to determine when the C2 state should be used instead of C3 state. While in the C3 state, the processor's caches maintain state but ignore any snoops.
Cache Line	The unit of memory that is copied to and individually tracked in a cache. Specifically, 64 bytes of data or instructions aligned on a 64-byte physical address boundary.
Cfg	Used as a qualifier for transactions that target PCI configuration address space.
Character	The raw data Byte in an encoded system (i.e., the 8b value in a 8b/10b encoding scheme). This is the meaningful quantum of information to be transmitted or that is received across an encoded transmission path.
CMI	IA-32 Core interface, Memory controller hub, I/O controller hub
Coherent (C)	Transactions that ensure that the processor's view of memory through the cache is consistent with that obtained through the I/O subsystem. In EP80579 integrated processor, Coherent (C) memory regions are coherent with IA caches when accessed from AIOC agents. Accesses to these memory regions enter the memory system through the IMCH. Memory regions that are coherent with IA caches must be accessible to the IA CPU.
Command	The distinct phases, cycles, or packets that make up a transaction. Requests and Completions are referred to generically as Commands.
Completion	A packet, phase, or cycle used to terminate a Transaction on a interface, or within a component. A Completion will always refer to a preceding Request and may or may not include data and/or other information.
Core Power Well	Main system power, turns off in S3 – S5
Cyclic Redundancy Check	A number derived from, and stored or transmitted with, a block of data in order to detect corruption. By recalculating the CRC and comparing it to the value originally transmitted, the receiver can detect some types of transmission errors.
Deasserted	Signal is set to a level that represents logical false.
Deferred Transaction	A processor bus Split Transaction. The requesting agent receives a Deferred Response which allows other transactions to occur on the bus. Later, the response agent completes the original request with a separate Deferred Reply transaction.
Delayed Transaction	A transaction where the target retries an initial request, but unknown to the initiator, forwards or services the request on behalf of the initiator and stores the completion or the result of the request. The original initiator subsequently reissues the request and receives the stored completion.
Direct Memory Access	Method of accessing memory on a system without interrupting the processors on that system.
Downstream	Describes commands or data flowing away from the processor-memory complex and toward I/O. The terms Upstream and Downstream are never used to describe transactions as a whole. (e.g. Downstream data may be the result of an Outbound Write, or an Inbound Read. The Completion to an Inbound Read travels Downstream.)

Table 1-4. Glossary Table (Sheet 3 of 5)

Term	Definition
Full Duplex	A connection or channel that allows data or messages to be transmitted in opposite directions simultaneously.
Gb/s	Gigabits per second (10^9 bits per second)
GB/s	Gigabytes per second (10^9 bytes per second)
Global visibility	An operation is said to be globally visible when all side-effects of the operation are visible to every observer in the system. For example, a write to some resource (e.g., memory location, control register, etc.) R achieves global visibility when a read of R by all other agents is guaranteed to return the new value.
Gx States	Global system states (Gx states) apply to the entire system and are visible to the user. • G3: Mechanical off - A computer state that is entered and left by a mechanical switch. It is implied by the entry of this off state through a mechanical means that no electrical current is running through the circuitry and that it can be worked on without damaging the hardware or endangering service personnel. • G2/S5: Soft Off - A computer state where the computer consumes a minimal amount of power. • G1: Sleeping - A computer state where the computer consumes a small amount of power, user mode threads are not being executed, and the system “appears” to be off (from an end user’s perspective, the display is off, and so on). • G0: Working - A computer state where the system dispatches user mode (application) threads and they execute. In this state, peripheral devices are having their power state changed dynamically.
Half Duplex	A connection or channel that allows data or messages to be transmitted in either direction, but not simultaneously.
Implicit Writeback	A snoop-initiated data transfer from the bus agent with the modified Cache Line to the memory controller due to an access to that line.
Inbound	A transaction where the request destination is the processor-memory complex and is sourced from I/O. The terms Inbound and Outbound refer to transactions as a whole and never to Requests or Completions in isolation. (e.g., an Inbound Read generates Downstream data, whereas an Inbound Write has Upstream data. Even more confusing, the Completion to an Inbound Read travels Downstream.)
Industry Standard Architecture	A 16-bit bus architecture associated with the IBM AT motherboard designed to connect motherboard circuitry to expansion card devices that is now considered Legacy.
Initiator	The source of requests. [IBA] An agent sending a request packet on 3GIO is referred to as the Initiator for that Transaction. The Initiator may receive a completion for the Request. [3GIO]
ISA Regime	A special legacy mode to support ISA-based devices which have been integrated into the chipset. It opens a dedicated channel from the peripheral device to the processor bus. While in this mode, the legacy device is granted exclusive accesses to memory and the ability to use Tenured Transactions.
Isochronous	A classification of transactions or a stream of transactions that require service within a fixed time interval.
Lane	A set of differential signal pairs, one pair for transmission and one pair for reception. A by-N Link is composed of N Lanes.
Layer	A level of abstraction commonly used in interface specifications as a tool to group elements related to a basic function of the interface within a layer and to identify key interactions between layers.
Legacy	Functional requirements handed down from previous chipsets, or PC compatibility requirements from the past.
Link	The collection of two Ports and their interconnecting Lanes. A Link is a dual simplex communications path between two components.
LPC Bus	Low Pin Count connection used to connect to the super I/O device.
Master	A device or logical entity that is capable of initiating transactions. A Master is any potential Initiator.
Mbyte/s	Megabytes per second (10^6 bytes per second)
Mem	Used as a qualifier for transactions that target memory space. (For example, a Mem read to I/O.)

Table 1-4. Glossary Table (Sheet 4 of 5)

Term	Definition
Metastability	A characteristic of flip flops that describes the state where the output becomes non-deterministic. Most commonly caused by a setup or hold time violation.
Multi Media Timer (MMT)	See High Precision Event Timer (HPET) in Table 1-3.
Non-Coherent	Transactions that may cause the processor's view of memory through the cache to be different than that obtained through the I/O subsystem.
North	Usually refers to bridges. The bridge or device that is closer to the processor-memory complex.
Ordering	Refers to the order in which signals and/or memory accesses to different locations must reach global visibility to ensure some behavior. Note that this excludes the "ordering" necessary to prevent data hazards which are accesses to the same location.
Outbound	A transaction where the request destination is I/O and is sourced from the processor-memory complex. The terms Inbound and Outbound refer to transactions as a whole and never to Requests or Completions in isolation. (For example, an Outbound Read generates Upstream data, whereas an Outbound Write has Downstream data. Even more confusing, the Completion to an Outbound Read travels Upstream.)
OWord	128 bits of data on a naturally aligned sixteen-byte boundary (e.g., the least significant four bits of the byte address are b"0000"). This is the native size of the IMCH datapath.
Packet	The indivisible unit of data transfer and routing, consisting of a header, data, and CRC.
PCI Reset	PCIRST#. This is the secondary PCI Bus reset signal. It is a logical OR of the primary interface PLTRST# signal and the state of the Secondary Bus Reset bit.
Peer-to-Peer	Transactions that occur between two devices independent of memory or the processor.
Platform Reset	IICH asserts PLTRST# to reset devices that reside on the primary PCI bus. The IICH asserts PLTRST# during power-up and when a hard reset sequence is initiated through the CF9h register. PLTRST# is driven inactive a minimum of 1 ms after both PWROK and VRMPWRGD are driven high. PLTRST# is driven for a minimum of 1 ms when initiated through the CF9h register.
Plesiochronous	From Greek, meaning almost synchronous. Describes signals that have the same nominal digital rate, but are synchronized on different clocks. Any variation in rate is constrained within specified limits, which allows a device to process the data signal without buffer underflow or overflow by making periodic compensating adjustments that repeat or delete dummy data bits. However, there is no limit to the phase difference that can accumulate between the signals over time.
Port	1. Logically, an interface between a component and a PCI Express* Link. 2. Physically, a group of Transmitters and Receivers located on the same chip that define a Link.
Posted	A Transaction that is considered complete by the initiating agent or source before it actually completes at the Target of the Request or destination. All agents or devices handling the Request on behalf of the original Initiator must then treat the Transaction as being system visible from the initiating interface all the way to the final destination. Commonly refers to memory writes.
Push Model	Method of messaging or data transfer that predominately uses writes instead of reads.
Queue	A first-in first-out (FIFO) structure.
Receiver	1. The Agent that receives a Packet across an interface regardless of whether it is the ultimate destination of the packet. 2. More narrowly, the circuitry required to convert incoming signals from the physical medium to more perceptible forms.
Request	A packet, phase, or cycle used to initiate a Transaction on a interface, or within a component.
Reserved	The contents or undefined states or information that are not defined at this time. Using any reserved area is not permitted. Reserved register bits must be set to 0. However, when stated, there may be specific instances where a reserved register is either non-zero, or there may be a requirement to make it non-zero.
Resume Power Well	Trickle from power supply, only turns off when power is disconnected from wall.
Resume Reset	Signal that resets the parts of the IICH in the resume power well, generated when the trickle supply turns on.
RTC Power Well	Powered by a coin cell battery and only turns off when the battery is drained. Powers the RTC and some resume events.

Table 1-4. Glossary Table (Sheet 5 of 5)

Term	Definition
Sx States	Sleeping states (Sx states) are types of sleeping states within the global sleeping state, G1. • S5: Soft Off state. The main memory power plane is shut down in addition to the clock synthesizer and core well power planes for the processor and CMI. The CMI resume well is still powered. • S4: Sleeping state - This state is only used to transition to or from the S5 state. **The S4 state is not a supported power management state in** CMI. • S3: Suspend to RAM (STR) state - The clock synthesizer and core well power planes for the processor and CMI are shut down, but the main memory power plane and the CMI resume well remain active. All clocks from synthesizers are shut down during the S3 state. • S0: Awake state - Power Management state when all power planes are active.
Simplex	A connection or channel that allows data or messages to be transmitted in one direction only.
SMBus	System Management Bus. A two-wire interface through which various system components may communicate.
Snooping	A means of ensuring cache coherency by monitoring all memory accesses on a common multi-drop bus to determine if an access is to information resident within a cache.
South	Usually refers to bridges. The bridge or device that is further from the processor-memory complex.
South Port	The PCI Express* downstream root port(s) on the IICH.
Split Lock Sequence	A sequence of transactions that occurs when the target of a lock operation is split across a processor bus data alignment or Cache Line boundary, resulting in two read transactions and two write transactions to accomplish a read-modify-write operation.
Split Transaction	A transaction that consists of distinct Request and Completion phases or packets that allow use of bus, or interconnect, by other transactions while the Target is servicing the Request.
Symbol	An expanded and encoded representation of a data Byte in an encoded system (e.g., the 10b value in a 8b/10b encoding scheme). This is the value that is transmitted over the physical medium.
Symbol Time	The amount of time required to transmit a symbol.
Target	A device that responds to bus Transactions. The agent receiving a request packet is referred to as the Target for that Transaction.
Tenured Transaction	A transaction that holds the bus or interconnect until complete, effectively blocking all other transactions while the Target is servicing the Request.
Transaction	An overloaded term that represents an operation between two or more agents that can be comprised of multiple phases, cycles, or packets.
Transaction Identifier	A multi-bit field used to uniquely identify a transaction. Commonly used to relate a Completion with its originating Request in a Split Transaction system.
Transmitter	1. The Agent that sends a Packet across an interface regardless of whether it was the original generator of the packet. 2. More narrowly, the circuitry required to drive signals onto the physical medium.
Upstream	Describes commands or data flowing toward the processor-memory complex and away from I/O. The terms Upstream and Downstream are never used to describe transactions as a whole. (For example, Upstream data may be the result of an Inbound Write, or an Outbound Read. The Completion to an Outbound Read travels Upstream.)
VCMI	IA-32 core, IA-32 Core interface, Memory controller hub, I/O controller hub

§ §

2.0 Architectural Overview

2.1 Overview

This chapter provides an overview of the Intel® EP80579 Integrated Processor product line architecture. Section 2.1.1, “Block Summary” gives a high-level summary for each of the major blocks and their internal interfaces. Section 2.1.2, “External Interfaces” reviews the EP80579’s external chip interfaces.

The block diagrams in Figure 2-1 and Figure 2-1 show the major EP80579 blocks.

2.1.1 Block Summary

- The EP80579 **IA-32 core** runs at 600, 1066, and 1200 MHz with an internal 400 or 533 MHz front-side bus (FSB) interface. The IA-32 core features a 256 Kilobyte 2-way level 2 cache (L2).
- The EP80579 **IMCH** provides the main path to memory for the IA-32 core and all peripherals that perform coherent I/O (e.g. PCI Express*, the IICH to coherent memory). The IMCH includes the four channel DMA engine as well as a PCI Express* root complex with 1x8, 2x4, or 2x1 interfaces. The **memory controller** operates at 200-266-333-400 MHz, depending on external DDR and SKU configuration. Depending on SKU, the EP80579 supports a single channel, 64-bit with ECC, memory controller for external DDR-2 memory (400, 533, 667, and 800 MHz). The EP80579 also supports a 32-bit with ECC mode for cost-sensitive applications.
- The EP80579 **IICH** provides a set of PC platform-compatible I/O devices that include two SATA1.0/2.0, one USB1.1/2.0 host controller supporting two USB ports, and two serial 16550 compatible UART interfaces. The IICH complex interfaces to the MCH through the “NSI” internal bus interface.
- The EP80579 **Acceleration and I/O Complex** (AIOC) supports three Gigabit Ethernet media access controllers, MDIO, Local Expansion Bus (LEB), two Controller Area Network (CAN) interfaces, IEEE1588 (2-GbE and 2-CAN ports), and SSP. In addition some SKU have three high-speed serial TDM interfaces that provide up to 12 T1/E1. ASU and SSU provide the high performance packet processing and accelerate common security capability.

Figure 2-1. Intel® EP80579 Integrated Processor Block Diagram

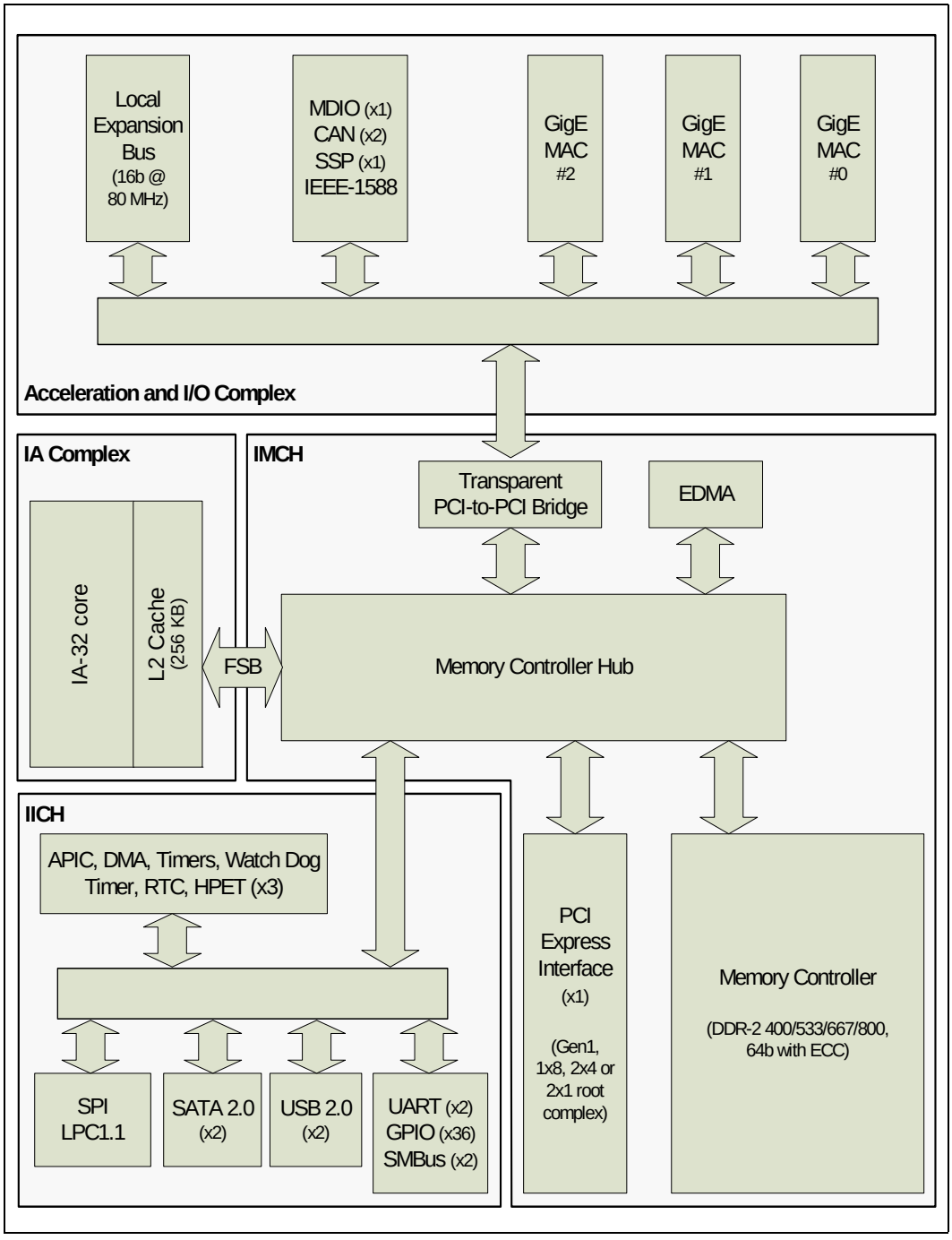
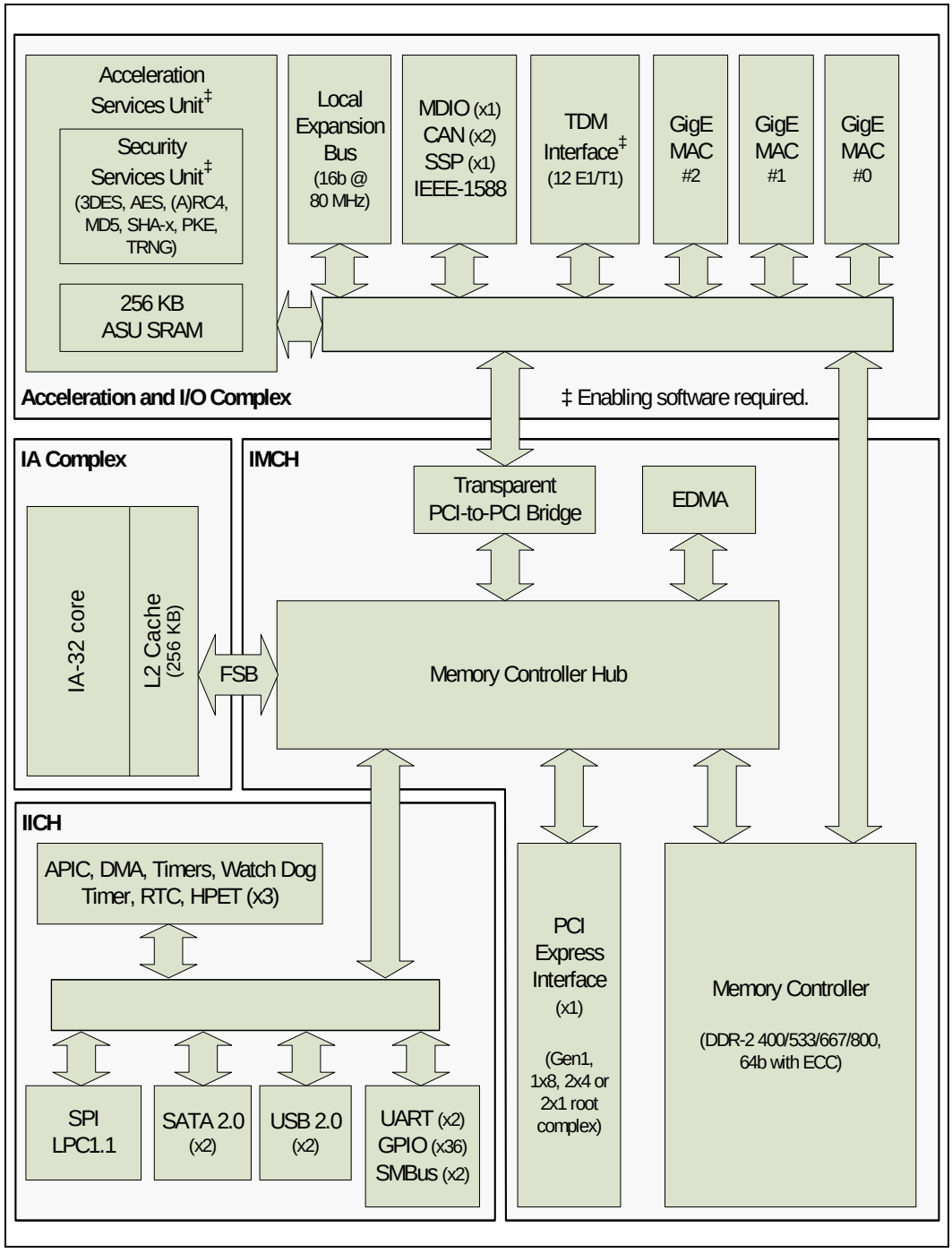


Figure 2-2. Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology Block Diagram



2.1.2 External Interfaces

Table 2-1 summarizes the key features of the external interfaces.

Table 2-1. EP80579 External Interface Summary

Name	Qty.	Description
External DDR Memory	1	Single channel memory controller with an ECC enabled 64-bit interface that, depending on SKU, supports external DDR-2 memory (400, 533, 667 and 800 MHz). Minimum memory size is 128 MB (in 32 bit mode). Maximum memory size is 4GB. The EP80579 integrated processor also supports a 32-bit mode (with ECC) for cost-sensitive applications.
PCI Express* (root)	1	Supports 1x8, 2x4, or 2x1 configurations as a root complex.
Gigabit Ethernet	3	10/100/1000 Gigabit Ethernet MACs with RGMII/RMII interface. Two of the three ports support IEEE 1588 hardware assist.
TDM	3	8.192 MHz high-speed synchronous serial TDM interfaces that support up to 12 T1/E1 links (SKU dependent) with Intel software driver provided for HDLC support.
Local Expansion Bus	1	25/16-bit 80MHz local expansion bus with 8 programmable chip selects.
USB 2.0	1	Universal Serial Bus 2.0 host controller interface, supports two USB ports (shared with USB1.1 ports)
USB 1.1	1	Universal Serial Bus 1.1 host controller interface, supports two USB ports (shared with USB2.0 ports)
LPC	1	Low Pin Count Bus (LPC) interface to attached PC compatible boot flash memory up to 64MB.
SPI	1	Serial Peripheral Interface (SPI).
GPIO	36	Programmable General Purpose I/O (GPIO) pins. Note, Intel recomends using this interface to boot from. Of the 36pins, many have alternate functions defined.
SMBus/I2C	2	I2C compatible SMBus2.0 connections.
UART	2	16550 compatible asynchronous serial ports that support data rate of at least 115Kbits/sec.
SATA	2	SATA1.0 or 2.0 used to attached external hard drives.
SSP	1	Synchronous Serial Port
CAN	2	Controller Area Network interfaces.
MDIO	1	MDIO interface to support the ethernet interfaces.
IEEE-1588	1	IEEE-1588 Hardware Assist

2.1.3 Frequencies and Gear Ratios

This section discusses frequencies and gear ratios.

The **IA core** used in the EP80579 is an IA-32 core, and can run at frequencies between 600 and 1200 MHz (actually supported CPU frequencies vary by SKU, see Chapter 47.0, "SKUs, Power Savings and Pre-Boot Firmware"). Based on divider ratios in its PLL design, the IA-32 core imposes a 6:1 minimum core/bus frequency ratio. This limits maximum FSB frequency to 400 or 533 MHz front-side bus (FSB) interface.

The IA-32 core/FSB frequency combinations supported are listed in Table 2-2.

Table 2-2. IA-32 core / FSB Frequency Ratios (depends on SKU and configuration)

IA-32 core [MHz]	FSB [MHz]	Ratio
600	400	6:1
1066	533	8:1
1200	533	9:1

Table 2-3 summarizes the memory controller clock ratios.

Table 2-3. Memory Controller Frequencies

DDR Clock [MHz]	DDR Technology
200	DDR2-400
266	DDR2-533
333	DDR2-667
400	DDR2-800

2.2 Signaling Architecture

As defined in Table 2-4, the EP80579 supports communication between the IA, AIOC devices (ASU, SSU, internal I/O devices) and externally attached bus master (PCI Express* through different memory types using a variety of operations.

Table 2-4. Summary of Communication

Operation	Agent	Targeted Memory Type[a]	
		External DRAM (Coherent)	External DRAM (AIOC Direct)
Read, Write	IA	X	X[b]
	ASU	X	X
	PCI-E DMA, EDMA	X	X[b]
	GbE	X	X
Atomic	IA	X	-
	ASU	X	X
	PCI-E DMA, EDMA	-	-
	GbE	-	-

a. "X" cells are supported with normal or MMIO accesses and "-" cells are not supported.

b. Agents in the IA-32 core may access AIOC-Direct memory (non-coherent) via uncacheable 1, 2 or 4-byte IA loads/store, IA MMX 8-byte MOVQ instructions or 64-byte DMA transfers. While such IA-32 core accesses to AIOC-Direct memory are not ordered with respect to other ASU memory traffic, the AIOC-Direct memory supports limited one-way communication in which IA agent (or the Internal PCI agent) is the sole writer and the internal PCI agent (or the IA agent) views such memory locations as read-only. Hardware will ensure that self-aligned 1,2, 4, 8 and 64-byte updates will be atomically visible to all readers.

The EP80579 allows signaling to occur between IA, AIOC complex, and externally-attached agents in the appropriate native signaling format.

The EP80579 supports basic producer/consumer behavior between agents.

Because the AIOC complex devices are exposed to the IA platform as PCI devices, they follow PCI ordering semantics when interacting with IA. This enables two fundamental producer/consumer models in coherent memory: the polled and interrupt methods.

For the Polled method:

1. Producer writes data to location “X” in Coherent Memory.
2. Producer sets flag to location “Y” in Coherent Memory
3. Consumer waits for flag to be set in Coherent Memory
4. Consumer read data from location “X” in Coherent Memory

To ensure this behavior, agents generating traffic into the IMCH must ensure that the writes originating from an agent are globally observable in the same order. In the above case X and Y must be globally observable in the same order.

For the Interrupt method:

1. Producer writes data to location “X” in Coherent Memory.
2. Producer generates an interrupt to Consumer (asynchronous signal)
3. Consumer reads interrupt status from Producer’s address space. Consumer waits for read to complete before issuing the next transaction.
4. Consumer reads data from location “X” in Coherent Memory

To ensure this behavior, an MMIO read issued to the PCI device (item 3 above) after a write that originated from this PCI device (item 1 above) must not complete out of order. The read completion must push ahead (flush) the write.

Also the MMIO read (item 3 above) should be to a location that is in the device. In the EP80579, it should not be to the PCI configuration registers but the device registers pointed to by the PCI BAR[1]. This ensures that IA device driver software for the EP80579 is not required to include explicit memory fence operations to enable producer-consumer synchronization for interrupt handling. Examples:

1. GigE ⇔ IA : GigE placing received data in coherent DRAM and interrupting IA, IA issuing a GigE CSR read, whose read completion must serialize the received DRAM data stream.
2. TDM ⇔ IA : TDM placing received data in coherent DRAM, then interrupting the IA, then IA gets a pointer to the data. An IA pointer dereference from the IA must see the TDM DRAM data.
3. ASU ⇔ IA : ASU placing data in coherent DRAM, then interrupting the IA, then IA gets a pointer to the data. An IA pointer dereference from the IA must see the ASU DRAM data.

1. The reason for this requirement is that the MCH config bus used to access the PCI configuration registers does not serialize the EP80579 internal PCI bus.

For communication using memory that is **not** coherent with the IA processor caches, or for direct communication between AIOC agents (e.g. between ASU, TDM, and GigE devices), the EP80579 may require software to insert explicit fencing operations to ensure correct producer/consumer behavior. Details are discussed in Chapter 4.0, "Signaling".

2.3 DMA and Peer-to-Peer Data Transfers

The EP80579 provides multiple DMA and DMA-like features that are summarized here:

- **IMCH EDMA Engine:** The four channel IMCH "Enhanced DMA" (EDMA) engine which supports DWORD aligned DMA. The EDMA engine supports memory-to-memory and memory-to-PCIe transfers. Read DMA granularity ranges from 1 byte to 4 Kilobytes. Write DMA granularity ranges from 1 to 256 bytes. The EDMA engine supports different source/target byte alignments that are important for packet processing, and is only programmable via IA PCI configuration space with completions signaled via IA interrupts. The EDMA engine can support burst data movement between "AIOC-direct memory" and IA coherent memory.
- **IICH DMA**: Supports IICH agents to/from memory, this is only used by USB and SATA.
- **LPC DMA:** Supports LPC agents. See Chapter 20.0, "LPC DMA".
- **No Peer-to-Peer Reads:** The IMCH does not support peer-to-peer reads.

Table 2-5 lists the supported DMA and peer-to-peer data transfer options.

Table 2-5. DMA and Peer-to-Peer Data Transfer Options

Usage Model#	Operation	Source	Destination	Initiator	Owner (Software)
1	PCI DMA read	DRAM	PCI Ex device	PCI Ex Device	IA device driver
2	PCI DMA write	PCI Ex Device	DRAM	PCI Ex Device	IA device driver
3	AIOC read	DRAM or AIOC device	AIOC device (GE)	Device	IA, ASU
4	AIOC write	AIOC device	DRAM or AIOC device (GE)	Device	IA, ASU
5	AIOC 3rd party read	DRAM	AIOC device (SSU)	ASU	ASU
6	AIOC 3rd party write	AIOC device (SSU)	DRAM	ASU	ASU
7	Peer-to-Peer PCI Read	Not Supported			
8	Peer-to-Peer PCI Write	AIOC Master or PCI Ex Device	PCI Ex Device only	AIOC Master or PCI Ex Device	AIOC Master Device Driver
9	Mem-to-Mem	DRAM	DRAM	EDMA Engine	IA
10	Mem-to-Memory Mapped IO	DRAM	PCI Ex or AIOC device	EDMA Engine	IA

§ §

3.0 Platform Memory and Device Configuration

3.1 Overview

This chapter presents the views of the major address spaces and device configuration structures as seen by various internal and external agents. Three related aspects are covered:

- The memory maps seen by various internal and external agents.
- The endianness seen by various agents and mechanisms the EP80579 uses to allow communication between agents with different endianness expectations.
- The PCI configuration infrastructure, which the EP80579 exposes through its memory maps.

3.1.1 Configuration Objectives

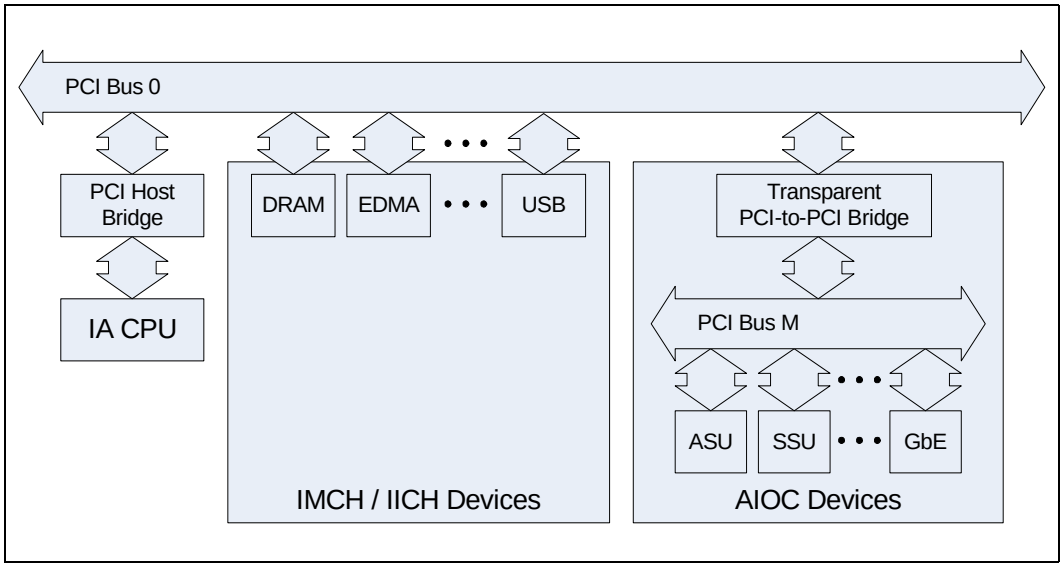
The EP80579 device and configuration model operates in a system-on-a-chip environment and blends the architectures of many disparate components into a unified whole. The major goals for the device configuration and access architecture include:

- Provide a configuration and access model that is aligned with existing IA platform algorithms.
- Support a unified address space model.

The IA-32 core is the primary agent responsible for device configuration. This is true across all supported SKUs.

To provide device configuration and operation capabilities that are aligned with the IA platform, the EP80579 uses the existing PCI infrastructure to expose on-die software-visible sub-blocks as devices on the PCI fabric. Figure 3-1 presents a logical overview of this organization.

Figure 3-1. Device-Centric Logical View of EP80579 Devices



The on-die IMCH and IICH devices materialize on PCI bus 0 while the AIOC devices materialize on PCI bus "M" that is behind a transparent PCI-to-PCI bridge on PCI bus 0 (the bus that the internal IMCH and IICH devices use). For simplicity, the figure does not show external devices.

3.1.2 Terminology and Conventions

Throughout this chapter, we will use the generic term "device" to refer to either a PCI device or a function of a PCI device. The text will be explicit when the distinction between device and function is important.

Addresses are always in hexadecimal and broken into 16-bit segments, for example, `0_FEED_BEEF`. When the distinction is important and not obvious, addresses are subscripted with "V", "P", or "S" for virtual, physical, or system address spaces, respectively.

The EP80579 addresses its DRAM in units of 8-byte quadwords. Before assigning byte addresses to the byte lanes in DRAM, we will refer to the locations as byte lane A through H as Table 3-1 illustrates.

Table 3-1. Main Memory DRAM Organization

Address	Byte Lane H	Byte Lane G	Byte Lane F	Byte Lane E	Byte Lane D	Byte Lane C	Byte Lane B	Byte Lane A
0	0H	0G	0F	0E	0D	0C	0B	0A
8	8H	8G	8F	8E	8D	8C	8B	8A
16	16H	16G	16F	16E	16D	16C	16B	16A

The byte located in address 0, lane A is referred to as 0A, the byte in address 0 lane B is 0B, etc.

3.2 IA Platform Infrastructure

The IMCH and IICH blocks (which, along with the IA-32 core, are collectively referred to as the VCMI) in the EP80579 provide an IA platform infrastructure with respect to endianness, address spaces and memory maps, configuration, etc.

This section focuses on the IA views and expectations around the endianness, address spaces and memory maps, and configuration for a basic IA platform. The EP80579 operates within this framework. These discussions highlight how and where the EP80579 differs from the framework. For additional detailed information on the IA infrastructure in the EP80579 specifically, see Section 9.0, "CMI Introduction", which discusses the IMCH implementation.

3.2.1 IA Platform View of Endianness

All memory in an EP80579 platform is little-endian to match requirements and expectations of an IA platform. The byte lanes (see Table 3-1 on page 112) are connected from the memory interface to the IMCH such that little-endian IA-32 core sees "byte 0" of a quad-word in memory in byte lane A and "byte 7" of a quad-word in memory in byte lane H.

Consider the following C code:

```
char c, *cp;                          // 1 byte

short s, *sp;                         // 2 bytes

long l, *lp;                          // 4 bytes

long long ll, *llp;                   // 8 bytes

cp = sp = lp = llp = (void *) 0x8;    // QW address 1

c  = *cp;

s  = *sp;

l  = *lp;

ll = *llp;
```

Executing this code on the IA-32 core in an EP80579 yields the following results:

```
c  == 8A

s  == 8B8A

l  == 8D8C8B8A

ll == 8H8G8F8E8D8C8B8A
```

Where the right-hand values in these results use the notation that Table 3-1 on page 112 in Section 3.1.2 describes to identify the byte lanes and address.

3.2.2 IA Platform View of Configuration

Because the IMCH and IICH blocks in the EP80579 come from an IA heritage, the EP80579 exposes much of the functionality in these blocks through a PCI infrastructure. The EP80579 extends this PCI infrastructure to expose the functionality in the AIOC, as Section 3.7, “PCI Configuration” on page 119 describes.

Before describing how the AIOC integrates with the IA-based IMCH and IICH blocks, it is helpful to consider how the PCI exposes IMCH and IICH functionality. Logically, the software-visible sub-blocks of the IMCH and IICH materialize as PCI devices and functions[1] on PCI bus 0 of the system through three independent address spaces:

- **Configuration Space:** Each function of each device has at least 256B of configuration space that is mapped to a fixed location by the platform (PCI Express* devices can provide for larger configuration spaces). This space provides system software with basic information on the device and allows for device-independent configuration.
- **Memory-Mapped I/O (MMIO) and I/O Spaces:** Each function of each device can request up to six MMIO and I/O regions of device-specified sizes to be mapped into physical address space through base address registers in the configuration header. System software selects the base address of each region. These spaces support device-specific operation such as access to device-specific control registers.

Of the thirty-two possible device slots on bus 0, five slots are reserved for software-visible blocks in the EP80579 IMCH and IICH and remainder are unused.

In general, the IMCH claims configuration accesses (i.e., those accesses that target configuration space) to device numbers 0, 1, 2, and 3 of bus 0 and routes configuration accesses to the remaining devices to the IICH over an internal NSI interface using Type 0 PCI configuration transactions[2] (see Section 13.2, “Platform Configuration Structure Conceptual Overview” and Section 13.3, “Routing Configuration Accesses”. In the EP80579 design, transactions to bus 0 devices that are sent through NSI to the IICH and unclaimed by the IICH will master abort.

Figure 3-2 presents a logical view of the EP80579 infrastructure for the software-visible blocks in the IICH and IMCH.

1. Except where the distinction is important, this document uses the term “device” to refer to both devices and functions in the PCI sense of these words.
2. Configuration transactions take the Type 1 form while in transit through the PCI fabric to their destination bus; upon reaching their destination bus, they become Type 0 transactions.

Figure 3-2. Logical Overview of the CMI PCI Infrastructure

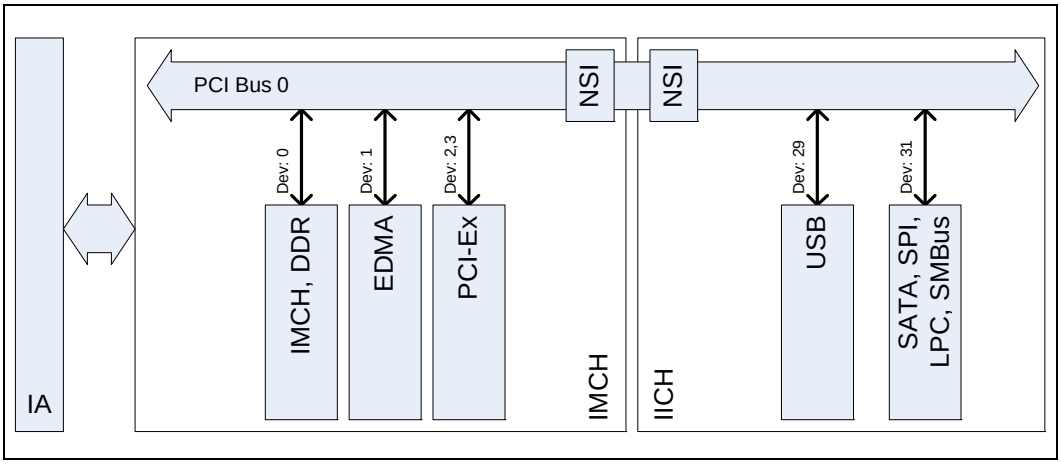


Table 3-2 summarizes the address space requirements (both in memory and I/O space) of the IMCH and IICH.

Table 3-2. Basic CMI Platform Address Space Requirements for IMCH and IICH Devices

Unit	PCI		Block	Mem, I/O	Size	Register
	Dev.	Fn.				
IMCH	0	0	Northbridge	Mem	4KB	SMRBASE (see Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register")
		1	DRAM, Error Handling	N/A	N/A	
	1	0	EDMA	Mem	4KB	EDMALBAR (see Section 16.3.1.9, "Offset 10h: EDMALBAR - EDMA Low Base Address Register"
	2	0	PCI-Express (HSI A0) ×8 or ×4	N/A	N/A	
	3	0	PCI-Express (HSI A1) ×4	N/A	N/A	
IICH	29	0	USB 1.1	I/O	32B	USBIOBAR (see Section 25.1.1.9, "USBIOBAR - Base Address Register")
		7	USB 2.0	Mem	1KB	MBAR (see Section 26.2.1.10, "Offset 10h: MBAR - Memory Base Address Register")

Table 3-2. Basic CMI Platform Address Space Requirements for IMCH and IICH Devices

	31	0	LPC/SPI	N/A	N/A	
		2	SATA	I/O	8B	PCMDBA (see Section 23.1.1.8, "Offset 10h: PCMDBA – Primary Command Block Base Address Register")
				I/O	4B	PCTLBA (see Section 23.1.1.9, "Offset 14h: PCTLBA – Primary Control Block Base Address Register")
				I/O	8B	SCMDBA (see Section 23.1.1.10, "Offset 18h: SCMDBA – Secondary Command Block Base Address Register")
				I/O	4B	SCTLBA
				I/O	16B	LBAR (see Section 23.1.1.12, "Offset 20h: LBAR – Legacy Bus Master Base Address Register")
				Mem	1KB	ABAR (see Section 23.1.1.13, "Offset 24h: ABAR – AHCI Base Address Register")
		3	SMBUS	N/A	N/A	

The IMCH and IICH devices in the EP80579 on PCI bus 0 allocate 18KB of memory space above and beyond the PCI L allocations in Figure 3-2 and 72B of I/O space through PCI BARs (this memory could be allocated in the "open" regions in Figure 3-2). This table does not include any memory regions that external devices (i.e., those attached to a IMCH PCI Express* port) or AIOC devices might allocate.

In addition to the IMCH and IICH devices, the EP80579 includes an IA-32 core that provides MSRs and other configuration structures.

3.3 High-Level Views

This section presents an overview of some of the general characteristics of the agents that the various EP80579 memory maps expose.

3.3.1 Characteristics of External System Memory (DRAM)

The address spaces in the EP80579 expose up to 4GB of physical system memory, in the form of DRAM, to be accessed by both on- and off-die agents.

Table 3-3. Memory Regions

Region	Managed By	Accesses to System Memory by AIOC Agents Coherent with IA Caches?	Contents
IA O/S	IA O/S	Y	IA O/S and application code and data structures
IA/ASU Shared (Coherent)	EP80579 Driver[a]	Y	IA and AIOC shared data structures
IA/ASU Shared (AIOC-Direct)	EP80579 Driver[a]	N	AIOC data structures; IA-32 core may access a portion via the EP80579 driver

a. The EP80579 Driver includes the EP80579-specific software stacks that run on the IA, ASU, etc.

Of the regions in Table 3-3, the "IA/ASU Shared (Coherent)" and "IA/ASU Shared (AIOC-Direct)" regions are ***not*** managed by the IA O/S. The EP80579 software expects that the BIOS carves this memory out of the memory map early in the boot process and sets it aside for use by the Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology software stack. As a result, the IA O/S does not allocate, manage, page, etc. these regions of memory.

The regions in Table 3-3 fall into one of two categories with respect to IA cache coherency: one that is coherent with IA caches for ***AIOC accesses*** and one that is not. It is important to note that the coherent/non-coherent category of a region affects ***only*** how the AIOC hardware handles a ***DRAM*** accesses. The category, in and of itself, does ***not*** have any implications on how IA must always access the region.

The EP80579 expects that, in general, all agents in the system can access all memory, consistent with their addressing capabilities, in the three regions that Table 3-3 lists. Exceptions to this general rule may arise due to the size of the address space that an agent supports or due to agent-specific aliasing of DRAM addresses onto other structures. The following sections on the memory maps outline any agent-specific exceptions. Finally, memory accesses that originate from the AIOC (or a device attached to the AIOC) must honor the coherency requirements in Table 3-3 based on the region they target. For consistency, software is expected to configure the EP80579 such that memory that the IA-32 core cannot access is not part of regions that are expected to be coherent with IA caches.

3.3.2 Characteristics of Internal and External Memories

Table 3-4 defines the supported operations by memory type. The table uses the following notation to indicate the behavior of the EP80579:

- "–" means the operation is not supported by the EP80579.
- "S" implies that the operation happens as a single atomic[1] update to memory. In other words, either the update is observable in its entirety or not at all.
- "M" implies that the operation may happen as multiple updates to memory. In other words, other agents can observe different parts of the affected memory location change values in any order but the end state of the memory location will be the desired value. This "flickering lights" effect makes such memory accesses useless for multi-agent synchronization unless a semaphore or flag variable is used to guard access to the shared location[2].

This table only applies to aligned-to-size operations; that is, a 4-byte operation is aligned to a 4-byte boundary, an 8-byte operation is aligned to an 8-byte boundary, etc.)

1. In the sense that it cannot be divided into multiple smaller writes.
2. Note that in guarding the location, visibility of the new flag must imply that the "flickering" has stopped.

Table 3-5. Supported Operations by Memory Type

Operation		IA-32 core		AIOC Agents	
Type	Size Bytes	Coherent (IA WB Cacheable)	AIOC-Direct[a]	Coherent	AIOC-Direct[a]
Read, Write	1	S	S	–	–
	2	S	S	–	–
	4	S	S	S	M
	8	S	S	S	S
	16	S	M	M	M
	32	–	–	M	M
	64	S	–	M	M
	128	–	–	M	M
Atom Read-Modify-Write (Semaphore)	1	S	M	–	–
	2	S	M	–	–
	4	S	M	S	S
	8	S	M	–	–

a. AIOC-Direct is a feature of the Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology SKU.

3.3.3 Characteristics of Device Configuration

To be able to leverage existing IA BIOS, O/S, and power management software, the EP80579's configuration mechanisms follow existing IA platform approaches. Of the four major components of the EP80579:

- The IA-32 core can use normal IA platform configuration algorithms.
- The IMCH and IICH can use normal IA platform configuration algorithms.
- To interoperate with normal IA platform configuration algorithms, the AIOC must be configured by the IA processor.

With the EP80579, the boot and configuration process is:

4. The IA boots from a FLASH device on the IICH SPI (LPC) interface. System software discovers and configures the devices on PCI bus 0 in the IMCH/IICH.
5. System software discovers and configures the devices behind the Transparent PCI-to-PCI bridge into the AIOC. This may amount to allocating memory regions specified by the BARs[1].
6. System software configures other buses on the system.

Once this process completes, the EP80579 is ready for operation.

It may be necessary for software to re-order the devices on bus 0 in the IMCH/IICH/AIOC to ensure that the AIOC can obtain the resources it needs. Since enumeration involves either a depth-first or a breadth-first traversal of the device tree from device zero (depending on the implementation of the platform PCI enumeration and discovery code), the system may not be able to honor a request for a large memory region from an AIOC device if this device is enumerated late in the process due to a large device number. This issue is not unique to the EP80579 and is handled in whatever fashion standard IA platform software handles such resource issues.

1. A 32-bit BAR can request a single memory size that is a power of 2 from 16B to 2GB according to the PCI specification.

3.4 Memory Map for IA-Attached Agents

There are two constraints in the EP80579 CMI and Memory Controller designs:

- The IA-32 core only supports 32-bit physical addresses.
- The Memory Controller supports at most 4GB of physical memory.

This discussion focuses on the perspective of an IA-attached agent; Section 3.5, "Memory Map for AIOC-Attached Devices" on page 119 provides similar discussion for AIOC-attached agents.

Table 3-6. Device Exposure from an IA-attached Memory Map Perspective

Device to Access	Materializes In Region	Notes
PCIe GigE MACs IMCH/IICH AIOC devices	PCI L	• PCI BAR(s) set by IA O/S or BIOS specify address mapping(s). • IA-32 core configures through I/O or PCI enhanced config spaces. • Reads, config space access between I/O agents not supported. • Region is at least 128MB per definition of TOLM, see Section 16.1.1.30, "Offset C4h: TOLM - Top of Low Memory Register".
DRAM	DRAM AD	• Contains IA/ASU Shared (AIOC-Direct) region from Table 3-3. • MENCBASE and MENCLIMIT registers define address range. • IA caches not coherent with AIOC accesses to this region. • Must include all DRAM that is inaccessible to the IA-32 core.
	DRAM C	• Contains I/A O/S, IA/ASU Shared (Coherent) regions from Table 3-3. • IA caches coherent with AIOC accesses to this region. • Cannot include any DRAM that is inaccessible to the IA-32 core.

3.5 Memory Map for AIOC-Attached Devices

AIOC-attached agents support several independent target IDs that provide independent address spaces. Table 3-7 summarizes, the addressing capabilities of AIOC agents range from 25 to 32-bits.

Table 3-7. Address Space Sizes of AIOC-attached Devices

Address Space Size [b]	Devices
32	Gigabit Ethernet MACs
32	ASU, SSU, TDM, SSP, CAN, 1588

3.6 Endianness

The EP80579 operates in an IA platform environment that is little-endian.

3.7 PCI Configuration

This section presents an overview of the implementation that integrates the AIOC and memory controller with the IA PCI infrastructure for configuration.

- PCI mechanisms (configuration space, memory-mapped I/O spaces, and I/O spaces) expose state for configuration.
- The IA-32 core performs all system configuration and initialization.

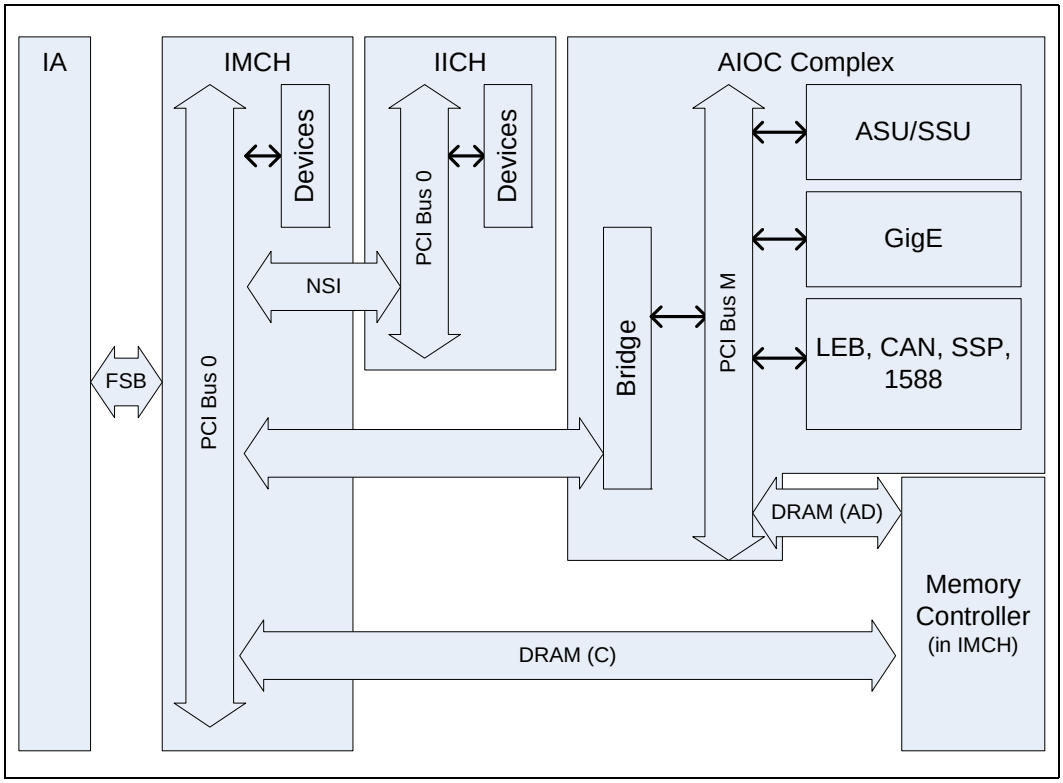
- The IA-32 core configures the AIOC, IMCH and IICH using standard IA platform algorithms (i.e., PCI-based discovery, enumeration, and configuration) with modifications for the specific mix of functionality that the EP80579 CPU, IMCH, and IICH instantiations provide.
- IA software configures external PCIe devices in the EP80579 using normal PCI discovery and configuration algorithms. The device and bus numbers for external devices are assigned by the BIOS and/or O/S during boot-time enumeration as normal.
- An EP80579-specific user driver handles interaction with external non-PCI agents attached to the EP80579 through its AIOC-side I/O interfaces. That is, the EP80579 user driver, not BIOS, will "discover" and operate any devices attached to the local expansion bus, for example. Note that since these devices do not implement PCI semantics, it is expected that they will not allocate MMIO regions beyond those already allocated for the PCI view of the appropriate AIOC device.
- The O/S always allocates an aperture in the memory map for any PCI device that defines one or more BARs, even if the device is unknown to the O/S at discovery.

The PCI abstraction that the AIOC provides for its devices is primarily a ***software*** abstraction for the purposes of configuration; the AIOC itself does ***not*** contain PCI devices.

3.7.1 Overview

The EP80579 integrates the AIOC and memory controller into the PCI fabric as Figure 3-3 describes. This figure presents a ***logical*** view of the system.

Figure 3-3. Attaching the AIOC to the CMI PCI Fabric (Logical Perspective)

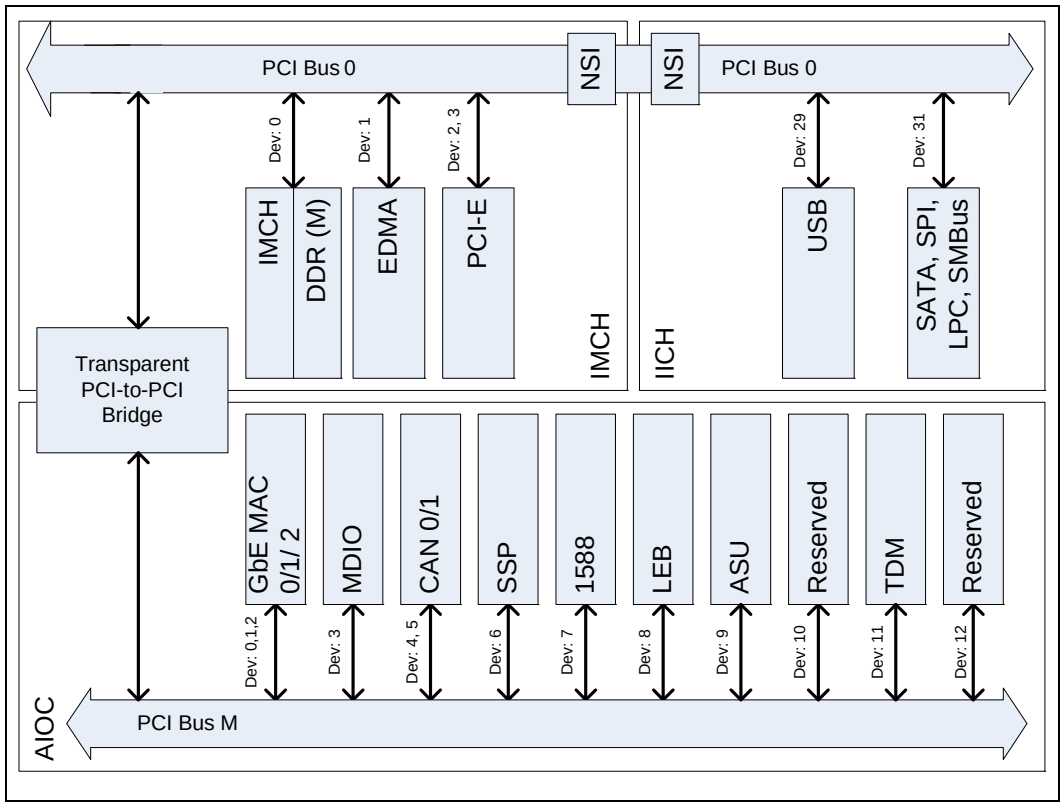


In this figure, PCI bus 0 originates in the IMCH and reaches internal IMCH PCI devices through internal paths. Bus 0 is bridged into the IICH via an NSI interconnect and into the AIOC. The memory controller materializes internally to the IMCH as device 0 of PCI bus 0. The internal transparent bridge materializes the devices for the AIOC through a bridged hierarchy as Figure 3-4 illustrates. In this hierarchy, a Transparent PCI-to-PCI bridge appear on PCI bus 0 with the remaining AIOC devices materializing behind the bridge on PCI bus M[1].

3.7.2 Device Tree

This section describes how the devices on the EP80579 die map onto the PCI device tree. In general, the EP80579 exposes the structures and sub-blocks that Section 3.7.1 describes through PCI devices that materialize behind a transparent bridge on bus 0. Figure 3-4 presents an overview of the device tree for on-die EP80579 software-visible sub-blocks (see also Figure 13-1, “Bus 0 Device Map” on page 348).

Figure 3-4. Overview of PCI infrastructure for On-die Devices



As mentioned earlier, AIOC devices materialize on bus “M” behind a bridge on PCI bus 0 where the IA BIOS or O/S assign the secondary bus number “M” at discovery.

Devices can request space in the system memory and I/O address maps through BARs in the configuration header. In general, the EP80579 materializes most device control and status registers in memory-mapped regions allocated by a BAR. The only exception

1. The IA BIOS and/or O/S assigns the specific bus number during PCI discovery and enumeration.

lies in the standard PCI configuration, status, and capability registers that PCI requires which materialize only in PCI configuration space. The AIOC devices that lie behind the bridge allocate by either the BIOS or OS to their regions of address and I/O space within a contiguous region of address space that the bridge claims. It is assumed that system software configures the aperture for the bridge to be large enough to cover all of the address space that the devices behind the bridge request.

The remainder of this section summarizes the device tree that the EP80579 implements. This summary includes a mapping between PCI devices and the EP80579 blocks along with the value of the device ID, class code, and a summary of the resources (i.e., registers, memory, etc.) that each device requests.

Table 3-9 summarizes the PCI devices that the IMCH and IICH materialize.

Table 3-9. IMCH and IICH PCI Device Summary

Device Name	EP80579 Units	PCI			SKU ID Number	Resources[a]
		B/D/F[b]	BSP[c]	Device ID		
Memory Controller Hub	IMCH	0 / 0 / 0	060000h	5020h	1,2,3,4,5,6, 7,8	MBAR (4KB and 4KB)
Error Reporting	IMCH	0 / 0 / 1	FF0000h	5021h	1,2,3,4,5,6, 7,8	
EDMA	EDMA	0 / 1 / 0	088000h	5023h	1,2,3,4,5,6, 7,8	MSI, MBAR (4KB)
PCI-Ex Port 0	PEA0	0 / 2 / 0	060400h	5024h	1,2,3,4,5,6, 7,8	PM, MSI
PCI-Ex Port 1	PEA1	0 / 3 / 0	060400h	5025h	1,2,3,4,5,6, 7,8	PM, MSI
PCI-to-PCI Bridge	IMCH	0 / 4 / 0	060400h	5037h	1,2,3,4,5,6, 7,8	
USB 1.1 Controller	USB1.1	0 / 29 / 0	0C0300h	5033h	1,2,3,4,5,6, 7,8	IOBAR (32B)
USB 2.0 Controller	USB2.0	0 / 29 / 7	0C0320h	5035h	1,2,3,4,5,6, 7,8	PM, MBAR (1KB)
LPC/SPI	LPC/SPI	0 / 31 / 0	060100h	5031h	1,2,3,4,5,6, 7,8	IOBAR (128B, 64B)
SATA	SATA	0 / 31 / 2	01018Ah, 010601h, 010401h, 010401h[d]	5028h, 5029h, 502Ah, 502Bh[e]	1,2,3,4,5,6, 7,8	PM, MSI, IOBAR (8B, 8B, 4B, 4B, 16B), MBAR (1KB)
SMBus	SMBUS	0 / 31 / 3	0C0500h	5032h	1,2,3,4,5,6, 7,8	IOBAR (32B)

a. "MBAR" is a memory space BAR, "IOBAR" is an I/O space BAR, "PM" is a power management capability, and "MSI" is an MSI capability.
b. PCI bus number, device number, and function number.
c. PCI base class code, subclass code, and programming interface PCI configuration register values.
d. SATA SC and PI values depends on the SATA mode and map value settings.
e. SATA DID value depends on the SATA mode settings.

Table 3-10. AIOC PCI Device Summary

Device Name	EP80579 Units	PCI			SKU ID Number	Resources[a]
		B/D/F[b]	BSP[c]	Device ID		
GigE MAC 0	GbE 0	M / 0 / 0	020000h	5040h	2,4,6,8	PM, MSI, ST, MBAR (128KB), IOBAR (32B)
				5041h	1,3,7,5	
				5042h	Reserved[d]	
				5043h	Reserved[d]	
GigE MAC 1	GbE 1	M / 1 / 0	020000h	5044h	2,4,6,8	PM, MSI, ST, MBAR (128KB), IOBAR (32B)
				5045h	1,3,7,5	
				5046h	Reserved[d]	
				5047h	Reserved[d]	
GigE MAC 2	GbE 2	M / 2 / 0	020000h	5048h	2,4,6,8	PM, MSI, ST, MBAR (128KB), IOBAR (32B)
				5049h	1,3,7,5	
				504Ah	Reserved[d]	
				504Bh	Reserved[d]	
MDIO	MDIO	M / 3 / 0	FF0000h	503Eh	1,2,3,4,5,6,7,8	PM, MBAR (4KB)
CAN Interface 0	CAN 0	M / 4 / 0	0C0900h	5039h	1,2,3,4,5,6,7,8	PM, MSI, ST, MBAR (4KB)
CAN Interface 1	CAN 1	M / 5 / 0	0C0900h	503Ah	1,2,3,4,5,6,7,8	PM, MSI, ST, MBAR (4KB)
SSP	SSP	M / 6 / 0	078000h	503Bh	1,2,3,4,5,6,7,8	PM, MSI, ST, MBAR (4KB)
IEEE 1588	1588	M / 7 / 0	111000h	503Ch	1,2,3,4,5,6,7,8	PM, MSI, ST, MBAR (4KB)
Local Expansion Bus	LE Bus	M / 8 / 0	068000h	503Dh	1,2,3,4,5,6,7,8	PM, MSI, ST, MBAR (4KB and 0-256MB[e])
ASU	ASU	M / 9 / 0	0B4000h	502Ch	2,4,6,8	PM, MSI, ST, MBAR (8KB, 16KB, 16KB, and 4KB)
				502Dh	1,3,7,5	
				502Eh	Reserved[d]	
				502Fh	Reserved[d]	
Reserved	Reserved	M / 10 / 0	088000h	503Fh	1,2,3,4,5,6,7,8	Reserved
TDM	TDM	M / 11 / 0	0B4000h	504Ch	1,3,5,7	PM, MSI, ST, MBAR (4KB and 4KB)
Reserved	Reserved	M / 12 / 0	110100h	5030h	1,2,3,4,5,6,7,8	Reserved

a. "MBAR" is a memory space BAR, "IOBAR" is an I/O space BAR, "PM" is a power management capability, "MSI" is an MSI capability, and "ST" is an EP80579 signal target capability.
b. PCI bus number, device number, and function number.
c. PCI base class code, subclass code, and programming interface PCI configuration register values.
d. These device IDs are reserved for future SKUs.
e. The size of the region that the 0-256MB MBAR requests is a multiple of 32MB between 0MB and 256MB (inclusive) based on reset-time platform configuration.

A summary of the registers, memory-mapped, and I/O-mapped resources that Table 3-9 and Table 3-10 identify can be found in Section 7.0, "Register Summary". Detailed descriptions of these resources can be found in the chapters that cover the relevant block.

3.7.3 Materializing Device Structures

The EP80579 exposes AIOC resources through standard PCI abstractions: configuration spaces, memory-mapped I/O spaces, and I/O spaces.

Access to MMIO and I/O spaces are accessed through memory and I/O read/write instructions, respectively. The addressing of these spaces for a given device depends on the specific mapping that the PCI configuration header establishes through BARs.

PCI defines two mechanisms for accessing the 256B of each device/function configuration registers located in PCI configuration space.

- PCI Mechanism: The header is accessed using 1-, 2-, or 4-byte `IN` and `OUT` instructions that access the PCI configuration address and data I/O ports at addresses `0CF8h - 0CFBh` and `0CFCh - 0CFFh`, respectively, in the IA I/O space. This mechanism allows access only to the 256B PCI-compatible configuration space
- PCI Express* Enhanced Mechanism: The header is accessed using 1-, 2-, or 4-byte memory accesses to the 256MB region starting at HECBASE ($0\_E000\_0000_P$ by default. This mechanism allows access to an expanded 4KB configuration space that PCI Express* defines (the first 256B are, by definition, the PCI-compatible configuration space).

These mechanisms differ in the address space they use to access the header. The PCI mechanism travels through IA I/O space while the PCI Express* Enhanced mechanism travels through IA memory space. The address format that the mechanisms use is identical to the standard IA platform format that encodes the PCI bus, device, and function numbers along with a register offset or number (see Section 13.6.0.1, “Offset 0CF8h: CONFIG_ADDRESS - Configuration Address Register” and Section 13.8.4, “Enhanced Configuration FSB Address Format” for details).

For either access method, the hardware in the AIOC that implements the configuration headers must be able to process accesses of the appropriate sizes.

3.7.4 PCI Configuration Headers

The PCI specification requires each PCI device to provide a 256B configuration space. The first 64B of this space contains a standard PCI configuration header and the remaining 192B contains any device-specific registers, capabilities records, etc. needed by the function. There are two flavors of configuration headers:

- All non-bridge devices provide a PCI type 0 configuration headers. This form of header is used to represent devices on the PCI fabric.
- All bridge devices provide a PCI type 1 configuration header. This form of header is used to represent bridge devices in the PCI fabric.

Because the AIOC devices are not PCI devices, they do not fully support all PCI configuration header fields[1]. The following tables describe the support in greater detail.

Table 3-11 summarizes the fields in a PCI type 0 header (i.e., header for non-bridge devices) and identifies which fields the EP80579 implements for AIOC devices. The EP80579 hardware implements the appropriate PCI semantics for all supported registers and fields in this table.

1. Configuration headers for IMCH and IICH devices follow PCI expectations as these devices are PCI compliant.

Table 3-11. PCI Configuration Header Support for Type 0 Headers in AIOC Devices (Sheet 1 of 2)

Offset	Register and Field		Bit(s)	Supt.[a]	Acc.[b]	Notes
00h - 01h	Vendor ID		15:0	Y	RO	Required by PCI.
02h - 03h	Device ID		15:0	Y	RO	Required by PCI.
04h - 05h	Command Register	Interrupt Disable	10	Y	RW	Supported in devices that can use INTx[c].
				N	RO	Not supported in devices that do not use INTx[d].
		Fast Back-to-Back Enable	9	N	RO	Not supported.
		SERR# Enable	8	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Parity Error Response	6	N	RO	Not supported. INTx/MSI signal AIOC errors.
		VGA Palette Snoop	5	N	RO	Not supported.
		Mem. Write & Inval. Enable	4	N	RO	Not supported.
		Special Cycles	3	N	RO	Not supported.
		Bus Master Enable	2	N	RW	ASU, GbE, TDM, LEB devices: Not supported[e], bit is implemented as RW but has no effect
				Y	RO	All devices except ASU, GbE, TDM, LEB: these devices cannot be bus masters.
		Memory Space Enable	1	Y	RW	All devices except 1588; these devices each materialize in memory space.
				N	RO	For IEEE1588; IEEE1588 does not materialize in memory space.
		I/O Space Enable	0	Y	RW	For GbE; GbE materializes in I/O space.
				N	RO	All devices except GbE; these devices do not materialize in I/O space.
06h - 07h	Status Register	Detected Parity Error	15	N	RO	Not supported; INTx/MSI signal errors.
		Signalled System Error	14	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Received Master-Abort	13	N	RO	Not supported[e].
		Received Target-Abort	12	N	RO	Not supported[e].
		Signalled Target-Abort	11	N	RO	Not supported; AIOC devices do not target-abort.
		DEVSEL Timing	10:9	N	RO	Not supported.
		Master Data Parity Error	8	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Fast Back-to-Back Capable	7	N	RO	Not supported.
		66MHz Capable	5	N	RO	Not supported.
		Capabilities List	4	Y	RO	Setup based on capabilities exposure by device.
		Interrupt Status	3	Y	RO	Supported in devices that can use INTx[c].
				N	RO	Not supported in devices that do not use INTx[d].
08h	Revision ID		7:0	Y	RO	Required by PCI.
09h - 0Bh	Class Code		23:0	Y	RO	Required by PCI.

Table 3-11. PCI Configuration Header Support for Type 0 Headers in AIOC Devices (Sheet 2 of 2)

Offset	Register and Field	Bit(s)	Supt.[a]	Acc.[b]	Notes
0Ch	Cache Line Size	7:0	N	RO	Not supported.
0Dh	Latency Timer	7:0	N	RO	Not supported.
0Eh	Header Type	7:0	Y	RO	Required by PCI.
0Fh	BIST	7:0	N	RO	Not supported.
10h - 27h	Base Address (x6)	6 x 31:0	Y	RW	Devices that materialize in I/O or memory spaces will populate these slots as necessary based on address space needs.
28h - 2Bh	CIS Pointer	31:0	N	RO	Not supported.
2Ch - 2Dh	Subsystem VID	15:0	Y	RO	Required by PCI.
2Eh - 2Fh	Subsystem ID	15:0	Y	RO	Required by PCI.
34h	Capability Pointer	7:0	Y	RO	Setup based on capabilities exposure by device.
3Ch	Interrupt Line	7:0	Y	RW	Supported in devices that can use INTx[c].
			N	RO	Not supported in devices that do not use INTx[d].
3Dh	Interrupt Pin	7:0	Y	RO	Supported in devices that can use INTx[c].
			N	RO	Not supported in devices that do not use INTx[d].
3Eh	Min_Gnt	7:0	N	RO	Not supported.
3Fh	Max_Lat	7:0	N	RO	Not supported.

a. Supported fields provide appropriate PCI semantics. Unsupported fields always return zero on reads unless otherwise noted.
b. RO and RW access types indicate that the register or field supports read-only access and read/write access, respectively.
c. AIOC devices that may signal via INTx include GbE, CAN, SSP, and IEEE1588.
d. AIOC devices that cannot signal via INTx include LEB.
e. This behavior is a deviation from the PCI specification for only the GbE and a device that can be a bus master.

Table 3-12 summarizes the fields in a PCI type 1 header (i.e., header for bridge devices) and identifies which fields the EP80579 implements. The EP80579 hardware implements the appropriate PCI semantics for all supported registers and fields in this table.

Table 3-12. PCI Configuration Header Support for Type 1 Headers in AIOC Devices (Sheet 1 of 3)

Offset	Register and Field		Bit(s)	Supt.[a]	Acc.[b]	Notes
00h - 01h	Vendor ID		15:0	Y	RO	Required by PCI.
02h - 03h	Device ID		15:0	Y	RO	Required by PCI.
04h - 05h	Command Register	Interrupt Disable	10	N	RW	Bridge-generated interrupts not supported.
		Fast Back-to-Back Enable	9	N	RO	Not supported.
		SERR# Enable	8	N	RW	Bridge delivers events that would be master aborts (primarily address decode) as SERR#.
		Parity Error Response	6	N	RW	Not supported. INTx/MSI signal AIOC errors.
		VGA Palette Snoop	5	N	RO	Not supported.
		Mem. Write & Inval. Enable	4	N	RO	Not supported.
		Special Cycles	3	N	RO	Not supported.
		Bus Master Enable	2	N	RW	Not supported[c].
		Memory Space Enable	1	N	RW	Bridge itself does not expose any non-standard registers via memory space.
		I/O Space Enable	0	N	RW	Bridge itself does not expose any non-standard registers via I/O space.
06h - 07h	Status Register	Detected Parity Error	15	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Signalled System Error	14	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Received Master-Abort	13	N	RO	Not supported[c].
		Received Target-Abort	12	N	RO	Not supported[c].
		Signalled Target-Abort	11	N	RO	Not supported; AIOC devices do not target-abort.
		DEVSEL Timing	10:9	N	RO	Not supported.
		Master Data Parity Error	8	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Fast Back-to-Back Capable	7	N	RO	Not supported.
		66MHz Capable	5	N	RO	Not supported.
		Capabilities List	4	Y	RO	Setup based on capabilities exposure by bridge.
		Interrupt Status	3	N	RO	Bridge-generated interrupts not supported.
08h	Revision ID		7:0	Y	RO	Required by PCI.
09 - 0Bh	Class Code		23:0	Y	RO	Required by PCI.
0Ch	Cache Line Size		7:0	N	RO	Not supported.
0Dh	Latency Timer		7:0	N	RO	Not supported.
0Eh	Header Type		7:0	Y	RO	Required by PCI.
0Fh	BIST		7:0	N	RO	Not supported.
10h - 17h	Base Address (x2)		2 x 31:0	N	RO	Not supported. Bridge does not expose non-standard registers via BAR.
18h	Primary Bus Number		7:0	Y	RW	Required by PCI.

Table 3-12. PCI Configuration Header Support for Type 1 Headers in AIOC Devices (Sheet 2 of 3)

Offset	Register and Field		Bit(s)	Supt.[a]	Acc.[b]	Notes
19h	Secondary Bus Number		7:0	Y	RW	Required by PCI.
1Ah	Subordinate Bus Number		7:0	Y	RW	Required by PCI.
1Bh	Secondary Latency Timer		7:0	Y	RO	Not Supported.
1Ch	I/O Base		7:0	Y	RW	Supported, AIOC devices materialize in I/O space.
1Dh	I/O Limit		7:0	Y	RW	Supported, AIOC devices materialize in I/O space.
1Eh - 1Fh	Secondary Status Register	Detected Parity Error	15	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Received System Error	14	N	RO	Not supported. INTx/MSI signal AIOC errors
		Received Master-Abort	13	N	RO	Not supported[c].
		Received Target-Abort	12	N	RO	Not supported[c].
		Signalled Target-Abort	11	N	RO	Secondary-side devices cannot target-abort.
		DEVSEL Timing	10:9	N	RO	Not supported.
		Master Data Parity Error	8	N	RO	Not supported. INTx/MSI signal AIOC errors.
		Fast Back-to-Back Capable	7	N	RO	Not supported.
		66MHz Capable	5	N	RO	Not supported.
20h - 21h	Memory Base		15:0	Y	RW	Supported, AIOC devices materialize in memory space.
22h - 23h	Memory Limit		15:0	Y	RW	Supported, AIOC devices materialize in memory space.
24h - 25h	Prefetch Memory Base		15:0	N	RW[d]	Prefetchable memory devices not supported.
26h - 27h	Prefetch Memory Limit		15:0	N	RW[d]	Prefetchable memory devices not supported.
28h - 2Bh	Prefetch Base (upper 32b)		31:0	N	RW[d]	Prefetchable memory devices not supported.
2Ch - 2Fh	Prefetch Limit (upper 32b)		31: 0	N	RW[d]	Prefetchable memory devices not supported.
30h - 31h	I/O Base (upper 16b)		15:0	N	RO	I/O spaces larger than 64KB not supported.
32h - 33h	I/O Limit (upper 16b)		15:0	N	RO	I/O spaces larger than 64KB not supported.
34h	Capability Pointer		7:0	Y	RO	Setup based on capabilities exposure by bridge.
38h - 3Bh	Expansion ROM Base		31:0	N	RO	Not supported.
3Ch	Interrupt Line		7:0	N	RO	Bridge-generated interrupts not supported.
3Dh	Interrupt Pin		7:0	N	RO	Bridge-generated interrupts not supported.

Table 3-12. PCI Configuration Header Support for Type 1 Headers in AIOC Devices (Sheet 3 of 3)

Offset	Register and Field		Bit(s)	Supt.[a]	Acc.[b]	Notes
3Eh - 3Fh	Bridge Control Register	Discard Timer SERR# Enable	11	N	RO	Not supported.
		Discard Timer Status	10	N	RO	Not supported.
		Secondary Discard Timeout	9	N	RO	Not supported.
		Primary Discard Timeout	8	N	RO	Not supported.
		Fast Back-to-Back Enable	7	N	RO	Not supported.
		Secondary Bus Reset	6	N	RW	Not supported.
		Master Abort Mode	5	Y	RO	Bridge delivers events that would be master aborts (primarily address decode) as SERR#.
		VGA Enable	3	N	RW	Not supported.
		ISA Enable	2	N	RW	Not supported.
		SERR# Enable	1	N	RW	Not supported.
		Parity Error Response	0	N	RW	Not supported.

a. Supported fields provide appropriate PCI semantics. Unsupported fields always return zero on reads unless otherwise noted and need not provide PCI semantics.
b. RO and RW access types indicate that the register or field supports read-only access and read/write access, respectively.
c. This is a known deviation from the PCI specification since the bridge can be a bus master.
d. This is a deviation from the PCI specification since this register should be RO on bridges that do not support prefetchable regions. The registers are RW for compatibility with the base IP. Software is expected to set these fields to indicate an empty region since no secondary-side devices (i.e., bus M) request prefethcable memory.

For additional details on the headers for AIOC devices, see Section 35.3.1, “Description of PCI Configuration Header Space”.

The specific portion of the 256B PCI configuration space that is active in a device depends on the needs of the specific device. In general, a device requires far less than 256B of storage to implement a typical configuration space. Regions of the 256B configuration space that are not required are reserved and need only support default behavior compliant with the PCI specification:

All PCI devices must treat Configuration Space write operations to reserved registers as no-ops; that is, the access must be completed normally on the bus and the data discarded. Read accesses to reserved or unimplemented registers must be completed normally and a data value of 0 returned.

§ §

4.0 Signaling

4.1 Overview

This chapter presents an overview of the inter-agent signaling mechanisms supported along with the implications on transaction ordering necessary to support producer/consumer software algorithms. This chapter concentrates primarily signaling for functional, not error, purposes while Section 5.0, “Error Handling” provides additional discussion on error signaling.

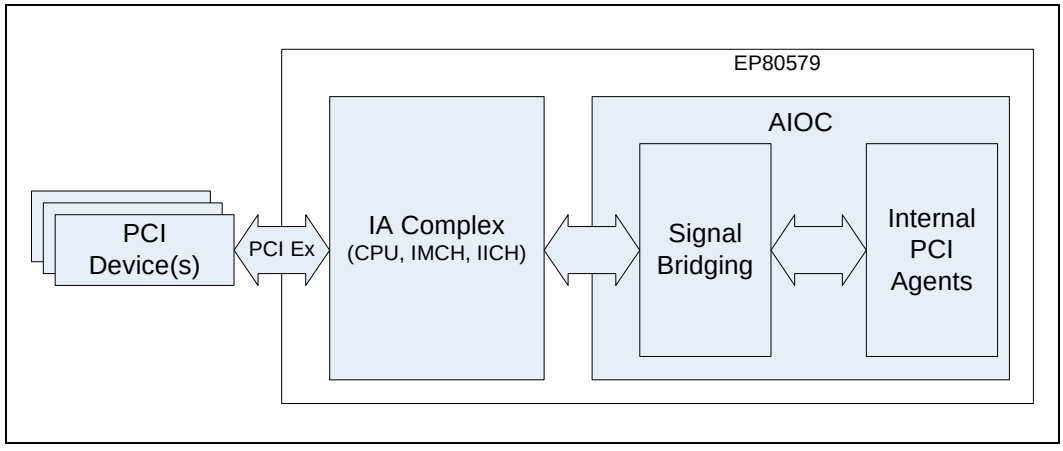
The EP80579 signaling model operates in a system-on-a-chip environment and must blend the signaling architectures of many disparate components into a unified whole.

- Allow signaling between IA agents (i.e., the IA-32 core, IMCH, IICH, and external PCI-Express-attached devices) and AIOC agents.
- Support producer/consumer relationships between agents in the IA or AIOC.
- Provide compatibility with existing IA signaling mechanisms for internal IMCH/IICH agents and external devices.
- Provide compatibility with IA platform signaling semantics and abstractions.

There are two tightly related parts to the signaling model:

- The *signaling mechanism* determines how takes a signal from a source agent, performs any necessary translation, and delivers it to a target agent in a form the target can understand.
- The *ordering mechanism* determines how orders a given signal from a source agent with respect to the data stream that the source agent produces to meet ordering requirements.

Figure 4-1. Logical Overview of Signaling Architecture



The IA complex attaches to the AIOC through a transparent PCI-to-PCI Bridge. Within the AIOC, the “Signal Bridge” block in Figure 4-1 represents the hardware in the AIOC, that converts signaling between the AIOC and IA domains and presents the appropriate abstractions to agents on either side of the bridge.

4.1.1 Terminology and Conventions

Throughout this section, we use the following terminology:

- **Global visibility:** An operation is said to be globally visible when all side-effects of the operation are visible to every observer in the system. For example, a write to some resource (e.g., memory location, control register, etc.) R achieves global visibility when a read of R by all other agents is guaranteed to return the new value.
- **Ordering:** Ordering refers to the order in which signals and/or memory accesses to different locations must reach global visibility to ensure some behavior. Note that this excludes the “ordering” necessary to prevent data hazards which are accesses to the same location.
- **Signal:** A message sent between agents to indicate some condition of interest. A signal may be an interrupt, a memory write, etc. This section uses the term when the specific transport medium is not important.

4.2 Existing Signaling Capabilities

There are several agents in the EP80579 that can be both the source and the target of a “signal”: the IA-32 core and accelerators in the AIOC. In addition, there are several other agents in the EP80579 that can be the source of a signal: devices in the IMCH or IICH, the error reporting hardware, the Gigabit Ethernet MAC, the SSP interface, the CAN interfaces, the IEEE 1588 interface, externally-attached PCI-Express devices, and externally-attached local expansion bus devices (via GPIO or INTx interrupts). This section summarizes the existing signaling capabilities of each of these agents.

When discussing signaling, this section classifies all signals into one of two general categories:

- *Ordered* signals must maintain a particular relationship with the data stream. For example, a MAC signals a CPU with an ordered signal after the MAC finishes writing inbound packet data to memory.
- *Unordered* signals need not maintain a particular relationship with the data stream. For example, memory interface hardware signals a CPU with an unordered signal when the interface encounters an uncorrectable memory error.

Signals of either type may participate in producer/consumer operations between agents; however, the operation can use only ordered signals if the producer/consumer operation involves the data stream[1]. Finally, note that this classification refers specifically to the data stream; signals may also be ordered with respect to other events (e.g., data being available in a local device buffer).

1. For producer/consumer operation to work correctly, software and hardware must be able to establish ordering relationships between various events in the operation. Such a relationship cannot be established between the data stream and a signal that is unordered with respect to the data stream.

4.2.1 IA-32 core/Platform

The IA-32 core supports two inbound signaling mechanisms that the EP80579 can use to accept signals from other agents: a legacy interrupt (INTx) and a message-signaled interrupt (MSI). The INTx mechanism encodes an interrupt on one of four out-of-band interrupt signals that drive interrupt controllers in the IA platform. The MSI mechanism encodes an interrupt as an in-band 32-bit write to a memory-mapped location. When the platform observes a write to an MSI location, it generates an interrupt to the CPU. The operating system or system software specifies, in large part, the data value that travels with an MSI; the device has limited ability to change or modify this value in the PCI MSI model.

The IA platform also supports signaling for errors through events that the platform eventually maps onto interrupts, such as SERR or SMI. Section 5.0, “Error Handling” discusses these mechanisms in further detail. These mechanisms are not used in functional inter-agent signaling.

4.2.1.1 MSI and INTx Signaling

The EP80579 supports both MSI and INTx mechanisms to interoperate with IA platform system software since not all operating systems or devices support MSIs. The PCI configuration header of each device indicates the capabilities of the device (i.e., if it can generate an MSI, which pin it uses in INTx mode, etc.) and also allows system software to specify the signaling mechanism the device should use when the device is capable of both MSI and INTx signaling. Further, when signaling the IA platform, software must tolerate “warts” of the IA signaling model such as spurious interrupts, etc.

For outbound signaling from the IA-32 core to other agents, the EP80579 relies on memory writes to MMIO locations to transport signals or IA can issue an MMIO read to a device CSR.

4.2.1.2 GPIO Signaling

The EP80579 provides the ability to configure a subset of its GPIO pins as interrupts. GPIO pins 16-21, 23-25, 27, 28, 30, 31, 33, 34, and 40 can function in either an interrupt mode or as a GPIO. Each of these GPIO pins can be connected to a single input on the APIC when software configures the GPIO as an interrupt. These pins are then available to external devices, such as a device attached to the Local Expansion Bus, for use as signals. The EP80579 handles signals arriving through a GPIO interrupt are handled like all other interrupts connected to the APIC. For additional information, see the GPIO material in Section 22.0, “General Purpose I/O: Bus 0, Device 31, Function 0”.

4.2.2 Other Agents

The remaining agents that are relevant to the EP80579 signaling model can only signal in the outbound direction; that is, they only generate signals from the agent into EP80579. These agents operate as follows:

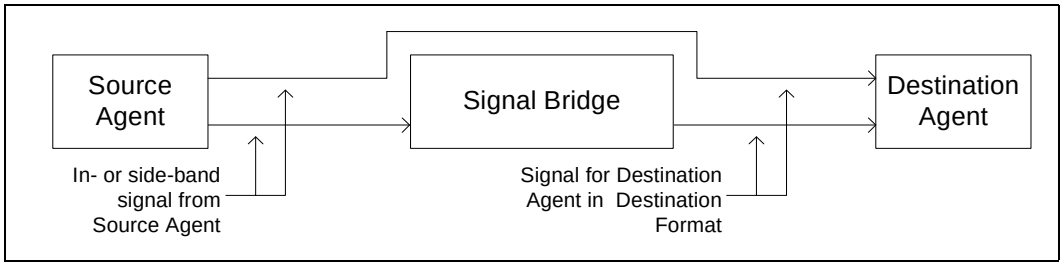
- IMCH, IICH, and externally-attached PCI devices generate IA platform INTx or MSI signals based on PCI device and platform configuration. PCI mechanisms such as SERR can also support error reporting (see Section 5.0, “Error Handling”).
- Gigabit Ethernet MAC generates a side-band interrupt signal.
- CAN, SSP, and IEEE 1588 Interfaces generate side-band interrupt signals.
- Externally-attached Local Expansion Bus devices generate side-band interrupt signals. These signals are not carried on the Local Expansion bus itself but rather are presented to the IA-32 core through GPIO pins that the system configures to generate signals.

4.3 Inter-Agent Signaling

Signals may originate from a number of sources within the system but may only target one of the on-die agents in the EP80579: the IA-32 core or AIOC accelerator. This section considers inter-agent signaling mechanisms available in the EP80579. This material looks at signaling in isolation.

Because the EP80579 integrates devices from several different fabrics, signaling between different agents may require bridge functionality to convert signals between the various existing capabilities that Section 4.2, "Existing Signaling Capabilities" on page 132 describes. Figure 4-2 presents a logical view of the flow of signals through the EP80579.

Figure 4-2. Logical View of Signaling Flow



In this model, a signal originates from a source agent as an in- or side-band signal. Depending on the specific situation, the signal may either reach the destination directly or through a signal bridge. The signal bridge in EP80579 does not provide any mechanism to ensure that a destination agent responds to a given inbound signal. As a result, software must be able to keep up with the signaling rates if catching and processing every signal is a requirement. If software is unable to keep up with the signaling rate, one or more signals can be dropped.

Table 4-1 summarizes the cross product of the combinations of producer and consumer agents that EP80579 supports for signaling between agents. This table identifies the types of signaling allowed as well as any bridging that the EP80579 hardware must provide to support signaling between the indicated agents.

Table 4-1. Supported Inter-Agent Signaling

Signal Bridging		On-Die Signal Consumer IA-32 core
Signal Producer	IA-32 core	Not Supported[a]
	IMCH, IICH Devices	PCI MSI / INTx via PCI
	GigE MAC	Side band: PCI MSI / INTx
	SSP	Side band: PCI MSI / INTx
	CAN	Side band: PCI MSI / INTx
	1588	Side band: PCI MSI / INTx
	Errors[b]	Side band: PCI MSI / INTx
	External PCI Express* Device	PCI MSI / INTx via PCI
	External Local Expansion Bus Device	MSI / INTx via GPIO

a. Support not required since there is only one such agent in an EP80579.
b. This includes signaling for errors in the memory controller, local expansion bus, etc.

In this table, ":" and "via" indicates bridged and direct paths for the signaling (see Figure 4-2), respectively. All EP80579 hardware that inter-operates with IA platform structures for signaling, such as the signal bridge, operate in IA physical address space. This allows it to access IA platform devices, such as the LAPIC that are accessed via MMIO reads and writes in the PCI L memory region.

The next two sections cover the various signaling scenarios that are represented in the cells of Table 4-1: signaling that travels around the bridge and that is bridged from a side band signal.

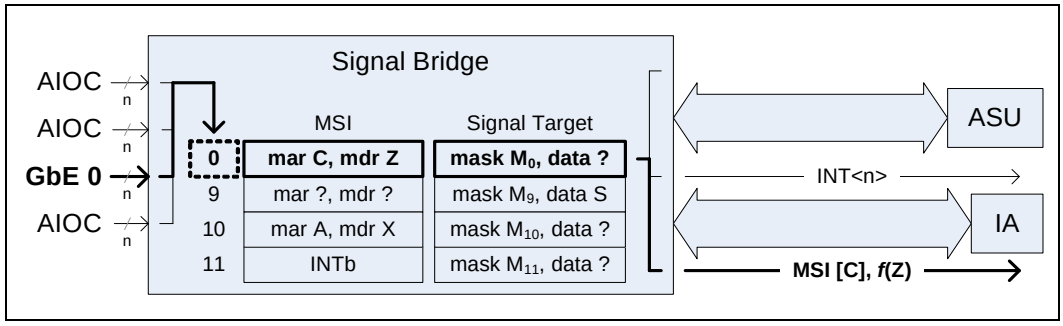
4.3.1 Signaling that Travels Around the Signal Bridge

For signaling that does not need to travel through the signal bridge, the EP80579 does not require any specific signaling support. In these cases, the signaling occurs over an existing path. For example, an interrupt from a IICH USB controller would travel over the existing IICH INTx or MSI path based on device configuration.

4.3.2 Signaling that is Bridged from a Side-Band Source Signal

The majority of signaling scenarios in Table 4-1 that require bridging are conversions between side band signals and a signal that targets the IA-32 core. The EP80579 uses a centralized agent, the signal bridge to perform this conversion. One or more side-band signals arrive at the signal bridge from each AIOC device that is capable of signaling. The bridge is responsible for generating the appropriate outbound signal to either an ASU device or the IA-32 core. Figure 4-3 presents an overview of the signal bridge hardware for a subset of the EP80579 AIOC devices.

Figure 4-3. Signal Bridging



Logically, the signal bridge consists of the MSI and Signal Target capability records from the PCI configuration headers for the AIOC devices. The MSI capability record for a device includes a message address register (mar in Figure 4-3) and Message data register (mdr in Figure 4-3) that indicate the address and data for the MSI as per the PCI definition of this capability. The Signal Target capability record for a device includes a mask that determines how the bridge steers the signal, data which identifies additional signaling data, and a status that indicates which side-band signal(s) has been asserted[1]. The Signal Target capability is a vendor-specific capability record whose format the EP80579 defines.

In the example show in Figure 4-3, side band signals arrive from Gigabit Ethernet MAC and other AIOC agents. These agents correspond to AIOC PCI devices. Here, the GbE agent generates a signal that the signal bridge delivers as an MSI to IA based on the

1. The status is not shown in Figure 4-3.

configuration in the MSI and Signal Target capability records in the PCI configuration header. These records allow the signal bridge to determine how to handle an inbound signal from a device.

All side band signals from a given source are collected at the signal bridge. For example, a source may provide interrupt lines and error condition lines that cause the EP80579 to send a signal when any one is asserted. Sets of side band signals are then associated with each AIOC PCI device that can signal. When a given side band signal asserts, the signal bridge looks up the MSI and Signal Target capability records in the PCI configuration header that corresponds to the device. These resources tell the signal bridge how and where to deliver the outbound signal:

- If the masks in the Signal Target capability record indicate the signal should be delivered to the IA-32 core, the signal bridge sends an MSI or INTx signal to the IA-32 core.
  - If the PCI configuration header selects MSI messaging, the signal bridge generates an MSI transaction from the device that generates the signal to the IA-32 core in accordance with the MSI capability record.
  - If the PCI configuration header selects INTx messaging, the signal bridge generates an INTx transaction to the IA-32 core.

The status register in the Signal Target Capability provides the state of the side-band signals associated with a given AIOC PCI device to help software disambiguate the source of the signal.

4.3.2.1 Targeting the IA-32 core with a Bridged Signal

To remain compatible with existing IA software stacks, EP80579 hardware supports the PCI MSI and legacy INTx signaling mechanisms into the CMI in response to signals targeting the IA-32 core.

- MSI enable field selects the signaling mechanism an MSI-capable device uses:
  - INTx legacy mode.
  - MSI mode.
- MSI capability record specifies how IA system software wants MSI-capable devices to signal the IA-32 core through a message address and message data register.
- Interrupt pin and line registers specify the interrupt pin and line that the hardware uses for legacy INTx mode.
- Interrupt disable bit in the command register specifies whether or not the device generates any signals.

Based on the PCI abstraction, these are ***per-device*** fields that are ***logically*** associated with a given device and that all devices that can signal the IA-32 core must implement.

Software is free to mix signal delivery mechanisms at the device level. For example, it may configure the EP80579 such that the Gigabit Ethernet MACs signal through MSIs while the remaining AIOC devices signal through INTx.

When device *X* signals the IA-32 core, it consults the PCI configuration header for device *X* to determine how to deliver the signal to IA (i.e., via MSI or INTx). For signaling via MSI, the header identifies the address and data value in the MSI transaction. Hardware builds the 32-bit data value for the MSI transaction from the contents of the PCI MSI Message Data Register, `mdr`, according to the PCI semantics. Specifically, the data value sent in the MSI to IA is:

- Bits 31:16 are zero.
- Bits 15:0 are `mdr`, the value of the PCI MSI Data Register in the PCI configuration header of the source device[1].

The address is given directly by the PCI MSI Message Address Register, `mar`. Finally, the source of the MSI transaction is set to the bus/device/function number of the device that generates the signal.

For signaling via INTx, the header identifies the interrupt line and pin that the EP80579 should use. In this case, hardware generates the appropriate transactions for INTx upstream into the CMI. Since the INTx mechanism cannot transport any information beyond the fact that a signal occurred, the EP80579 needs to expose enough device state to the software stack, via device-specific control register(s), to allow software to be able to determine both the source and cause of the interrupt. Such state would be in addition to the generic interrupt state that each PCI device provides through its PCI configuration header as per the PCI specification

To integrate with the existing IMCH/IICH, the signal bridge in the EP80579 will generate four INTx signals that it tracks based on the interrupt state of the blocks from the AIOC that can generate signals. These four INTx signals are provided to interrupt hardware in the IA platform hardware where they are ORed with similar signals from other agents and converted into the appropriate signaling to the IA-32 core. This hardware will also provide a signal back to the IMCH that indicates when the local INTx state can be deasserted.

In addition to generating any transaction(s) necessary to send the signal to the IA, the AIOC must preserve the semantics of PCI interrupts and signals with respect to the state in the PCI configuration headers for the AIOC devices.

- Signaling to the IA-32 core by a device should operate in accordance with the MSI mode and configuration in the MSI capability record.
- Signaling to the IA-32 core by a device must be disabled when the interrupt disable bit in the PCI command register is set
- The interrupt status bit in the PCI status register should reflect the status of an INTx signal.

This preserves the PCI abstraction for AIOC devices.

§ §

1. This assumes that the device requests exactly one message in the PCI MSI capability record [PCI_3].

5.0 Error Handling

5.1 Overview

This section presents an overview of the error handling mechanisms that the EP80579 provides. The intent of this discussion is to provide a broad background to error handling on the chip. Register definitions and other error handling details can be found in the discussions of the relevant units throughout this document.

5.2 EP80579 View of Error Reporting

This section describes the guiding principles on which the error logging and reporting registers are based for the EP80579. For the purposes of this discussion, an “error” is an exceptional condition that is beyond the control of the EP80579 hardware or software and might result in data corruption or data loss. This definition does not cover “functional” errors that are not beyond the control of the EP80579. For example, this section discusses double-bit ECC errors as they involves data corruption that occurs through no overt action of the EP80579; it does not cover an underflow error on a ring since this is a functional condition that can be managed in software to avoid data loss.

5.2.1 Hardware Capabilities

With respect to their error handling capabilities, the blocks on the EP80579 can be divided into three groups: the IA blocks (including the IA-32 core, IMCH exclusive of the memory interface, IICH), the memory controller interface, and the AIOC blocks. Generally speaking,

- The IA blocks (see Section 5.3, “Error Reporting by the IMCH” on page 141 and Section 5.4, “Error Reporting by the IICH” on page 149) use the FERR/NERR and PCI error reporting infrastructures.
- The memory controller block (see Section 5.5, “Error Reporting by the System Memory Controller” on page 153) uses the FERR/NERR error reporting infrastructure.
- The AIOC blocks (see Section 5.6, “Error Reporting by AIOC Devices” on page 155) use their existing error reporting infrastructures that are “bridged” into the IA-32 core to report errors to IA through PCI signals (i.e., INTx or MSI depending on device configuration). These blocks do not support other PCI error reporting capabilities such as SERR or the IMCH FERR/NERR architecture.

This organization allows an EP80579 system to use standard IA platform reporting abstractions and algorithms throughout the IA portion of the chip (including the memory controller). AIOC devices then use PCI INTx or MSI signaling to present their error handling within the IA infrastructure. Utilizing signaling in this fashion implies that the responsibility for error handling in AIOC devices resides with Intel provided AIOC device drivers in the EP80579 software stack.

The EP80579 hardware, where appropriate, supports two general capabilities for error handling. The first is support for data “poisoning” to propagate errors through the chip. The second is support for mechanisms to allow software to inspect the cause of the error.

To allow software to inspect the error, each unit in the EP80579 that is capable of detecting errors typically provides hardware support to:

- Log sufficient information for software to determine what the error was along with relevant details on the error.
- Disable error detection and reporting.
- Report if more than one error occurred. At a minimum, the unit will note that a second error occurred. Optionally, a unit may gather additional information on subsequent errors such as the type(s) of errors or other relevant details.

A given unit need not support all of these capabilities.To illustrate these capabilities, consider the EP80579 DRAM interface. DRAM on the EP80579 is protected by ECC, so, on a read with a double-bit error, the memory controller reports an error. The memory controller can poison the data return value to inform future consumers of the data that the data is bad.

For software, registers in the memory controller provide the ability to mask this error along with logging registers that captures the address that was being read when the error was encountered. On an unmasked double-bit error, the logging register captures the address and the memory controller signals the IA-32 core through the FERR/NERR infrastructure. The error address register is locked at this point, so subsequent errors will not overwrite it until software handles the error and unlocks the registers by writing a register to clear the error condition. The memory controller provides a “next error” register that can capture information on other unmasked errors that occur before software clears the double-bit error condition.

In general, the CMI will attempt to route requests based on their understanding of the address space layout of the platform (that is, the amount of installed physical DRAM, attached PCI Express* devices, etc.). There are several tables throughout the EAS that define how various parts of the chip handle this routing task:

- Section 10.1, “Overview” and Section 10.2, “IMCH Responses to EDMA Transactions” describes how the IMCH responses to transactions from the EDMA engine.
- The memory controller does not perform bounds checking on addresses, error handling behavior is determined by the upstream agents that pass the request to the memory controller as Section 5.5.1, “Handling Out-of-Bounds Addresses” on page 154 describes.

The IA portion uses the standard IA mechanisms to handle cases where this routing encounters errors (e.g., accessing an unpopulated region of memory). These cases cause aborts and are escalated through the normal error handling paths. On errors in requests arriving from the memory target, the IMCH will drop writes (i.e., not forward them into the IMCH) and poison data returns for reads through the appropriate push/pull data error signals. These error conditions are reported through IMCH error reporting registers.

5.2.2 Software Usage Model

The software responsibilities for error handling are split between BIOS and system/ application software. The manner in which the software uses the capabilities that the hardware provides depends on the specific software stack under consideration.

In general, BIOS does not establish a usage model for error handling. Its primary responsibility is to configure the error handling registers throughout the EP80579 in a manner consistent with the needs of system and application software. The EP80579 reference BIOS provides this level of support. However, in general, a BIOS implementation may also log information on error events reported via SMI[1]. In this case, BIOS populates data structures in SMBIOS[2] that an operating system can later retrieve.

Generally speaking, the software is more interested in seeing an error rather than the specific location where it is sent. The usage model from the system or application perspective depends on the manner in which the hardware reports the event to the system along with the specific error event. There are two general types of reporting that one can consider: through a kernel-trapped signal and through a non-kernel-trapped signal. A particular error event need be reported through at least one of these mechanisms.

When the hardware reports an error event through a signal that the OS kernel handles on its own (e.g., MCERR, SERR, etc.), the operating system takes whatever corrective action it implements for the signal: bug check, panic, halt, reboot, event logging, clean up and continue, etc. In this case, the OS kernel establishes the usage model. This approach is only applicable to the IA and memory controller blocks since the AIOC devices can only report error events through INTx or MSI signals that the OS kernel passes off to the appropriate driver software rather than handle in the kernel.

When the hardware reports an error event error through a signal that the software outside of the OS kernel handles on its own (e.g., INTx, MSI, etc.), it is the responsibility of driver or other non-kernel software to take corrective action. All AIOC agents fall into this category since their hardware uses only INTx or MSI for error reporting. In this case, it is the responsibility of the driver software to establish the error handling usage model. Typically, the action the driver takes in response to an error event will match those taken by the OS kernel: panic, clean up and continue, etc.

5.3 Error Reporting by the IMCH

See Section 14.2, “Exception Handling” and Section 14.3, “Error Conditions Signaled” for further discussion on IMCH error handling.

5.3.1 Overview of the First and Next Error Architecture

The IMCH provides a “first” and “next” error architecture wherein errors accumulate locally in unit-level first/next error registers that the IMCH aggregates into global first/ next error registers. Once a unit records an error event in its “first” error register, it will record all subsequent errors in its “next” error register until software clears the error condition in the first error register. The error events are then classified into “fatal” and “non-fatal” groups for reporting through the global first and next error registers. The architecture allows software to mask individual error events at the unit level. In addition, through per-unit registers, software can configure the hardware to report the error event through IA SMI, SCI, SERR, or MCERR signals.

1. This approach only works for error events that the EP80579 can report through SMI. Specifically, this approach would not work for error events from AIOC agents which cannot report through SMI in the EP80579.
2. System Management BIOS (SMBIOS) is a specification to lay out data structures and access methods in a BIOS which provides for storage and retrieval of information about the PC in question

The actions taken by the hardware in response to an error event depends on the manner in which software has configured the IMCH to report the error. For unmasked events that the IMCH signals through SMI, SCI, or SERR, the IMCH presents the error to the IA-32 core indirectly through the IICH. In these cases, the IMCH sends an error message to the IICH (via NSI) that IICH interrupt logic handles by signaling the IA-32 core through an interrupt. For unmasked events that the IMCH signals through MCERR, the IMCH directly presents the error to the IA-32 core (via FSB). In this case, the IMCH directly signals the IA-32 core through the MCERR protocol on the FSB.

After receiving the message, software will query the global IMCH error registers (see Section 5.3.2, "Global Error Events") to determine which IMCH unit is responsible for the error. With that information, software can query the unit that generated the event to determine the specific cause.

5.3.2 Global Error Events

Table 5-1 summarizes the error events that the IMCH captures in its GLOBAL_FERR and GLOBAL_NERR error registers (see Section 16.2.1.12, "Offset 40h: GLOBAL_FERR - Global First Error Register" and Section 16.2.1.13, "Offset 44h: GLOBAL_NERR - Global Next Error Register"). Each of these events rolls up one or more unmasked error events from an individual IMCH unit. To determine the specific error event that causes a signal, software consults the unit-specific error registers that Table 5-1 indicates. The global registers provide summary information only; masking takes place at the unit level.

Table 5-1. Summary of IMCH Global Error Conditions

Event	Fatality[a]	Unit-Specific Registers	Notes
DRAM Controller Fatal Error	Fatal	DRAM_FERR, DRAM_NERR	Fatal error in DRAM interface.
FSB Fatal Error	Fatal	FSB_FERR, FSB_NERR	Fatal error on internal CPU/IMCH FSB interface.
NSI Fatal Error	Fatal	NSI_FERR, NSI_NERR	Fatal error on internal IMCH/IICH NSI interface.
DMA Fatal Error	Fatal	EDMA_FERR, EDMA_NERR	Fatal error from DMA controller.
PCI Express* Port A1, A0 Fatal Error	Fatal	PEAFERR, PEANERR[b]	Fatal error from PCI Express* Port A1 (PEA1) or A0 (PEA0).
Buffer Unit Non-Fatal Error	Non-Fatal	BUF_FERR, BUF_NERR	Non-fatal error in posted memory write buffer.
DRAM Controller Non-Fatal Error	Non-Fatal	DRAM_FERR, DRAM_NERR	Non-fatal error in DRAM interface.
FSB Non-Fatal Error	Non-Fatal	FSB_FERR, FSB_NERR	Non-fatal error on internal CPU/IMCH FSB interface.
NSI Non-Fatal Error	Non-Fatal	NSI_FERR, NSI_NERR	Non-fatal error on internal IMCH/IICH NSI interface.
DMA Non-Fatal Error	Non-Fatal	EDMA_FERR, EDMA_NERR	Non-fatal error from DMA controller.
PCI Express* Port A1, A0 Non-Fatal Error	Non-Fatal	PEAFERR, PEANERR[b]	Non-fatal error from PCI Express* Port A1 (PEA1) or A0 (PEA0).

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Each port has its own independent PEAFERR and PEANERR registers in the PCI configuration space for the port controller device.

Section 5.3.3, Section 5.3.4, Section 5.3.5, Section 5.3.6, Section 5.3.7, and Section 5.3.8 discuss the specific per-unit events that the IMCH hardware captures and rolls up into the global conditions that Table 5-1 lists.

5.3.3 Unit-Level Errors from the Buffer Unit

The IMCH buffer unit captures error events from the memory system coherent Posted Memory Write Buffer (PMWB) in the BUF_FERR and BUF_NERR registers. The buffer unit reports an error event to the IA-32 core through SCI, SMI, SERR, or MCERR signals based on the settings in the BUF_SCICMD, BUF_SMICMD, BUF_SERRCMD, and BUF_MCERRCMD registers. Software can independently configure the specific signal that each buffer unit error event uses.

Table 5-2 summarizes the error conditions that the PMWB can generate.

Table 5-2. Summary of IMCH Buffer Unit Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
DRAM to PMWB Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error detected on read from DRAM agent by PMWB.
System Bus or I/O to PMWB Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error detected on write to PMWB from system bus or I/O agent.
PMWB to System Bus Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error detected on data to the system bus.
PMWB to DRAM Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error detected when PMWB is flushed to DRAM.

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on BUF_SCICMD, BUF_SMICMD, BUF_SERRCMD, and BUF_MCERRCMD register values.

Table 5-3 summarizes the capabilities of the IMCH buffer unit error handling for each of the features that the unit is expected to provide.

Table 5-3. Summary of IMCH Buffer Unit Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The BUF_EMASK (see Section 16.2.1.29, "Offset 74h: BUF_EMASK - Memory Buffer Error Mask Register"), BUF_SCICMD, BUF_SMICMD, BUF_SERRCMD, and BUF_MCERRCMD registers enable and mask error reporting. The PCICMD register (see Section 16.2.1.3, "Offset 04h: PCICMD - PCI Command Register") also enables and masks SERR signals.
Logging Details	IMCH does not capture error logging information beyond the event flags in the BUF_FERR, BUF_NERR and PCISTS (see Section 16.2.1.4, "Offset 06h: PCISTS - PCI Status Register") registers.
Reporting Multiple Errors	The BUF_NERR register captures "next" errors. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	IMCH passes along error information to poison data.

5.3.4 Unit-Level Errors from the DRAM Interface

These errors include the error events reported by the memory controller, see Section 5.5, "Error Reporting by the System Memory Controller" on page 153 for additional details.

5.3.5 Unit-Level Errors from the FSB Interface

The FSB interface captures error events from the FSB interface that connects the IA-32 core to the IMCH in the FSB_FERR and FSB_NERR registers. The FSB interface reports an error event to the IA-32 core through SCI, SMI, SERR, or MCERR signals based on the settings in the FSB_SCICMD, FSB_SMICMD, FSB_SERRCMD, and FSB_MCERRCMD registers. Software can independently configure the specific signal that each buffer unit error event uses.

Table 5-4 summarizes the error conditions that the FSB can generate.

Table 5-4. Summary of IMCH FSB Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Outgoing I/O Data Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error on outgoing data from I/O subsystem.
Outgoing Memory Data Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error on outgoing data from memory subsystem.
FSB BINIT# Detected	Uncorrectable	Fatal	N/A[c]	Electrical high-to-low transition of BINIT#.
FSB MCERR# Detected	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Electrical high-to-low transition of MCERR# when CMI is not driving.
Non-DRAM Lock Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Lock detected to memory space that does not map to DRAM.
FSB Addr. Above TOM/TOLM	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Address detected above TOM/TOLM.
FSB Data Parity	Uncorrectable	Non-Fatal	N/A[c]	Parity error on FSB detected.
FSB Addr. Strobe Glitch Detected	Uncorrectable	Fatal	N/A[c]	Glitch detected on FSB address strobe.
FSB Data Strobe Glitch Detected	Uncorrectable	Fatal	N/A[c]	Glitch detected on FSB data strobe.
FSB Request/Addr Parity	Uncorrectable	Fatal	N/A[c]	Party error on FSB address or request signals.

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on FSB_SCICMD, FSB_SMICMD, FSB_SERRCMD, and FSB_MCERRCMD register values.
c. Although the IMCH supports these errors, the EP80579 will not ever generate them since its on-die FSB implementation does not support BINIT# or parity.

Table 5-5 summarizes the capabilities of the FSB error handling for each of the features that the unit is expected to provide.

Table 5-5. Summary of IMCH FSB Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The FSB_EMASK FSB_SCICMD, FSB_SMICMD, FSB_SERRCMD, and FSB_MCERRCMD registers enable and mask error reporting. The PCICMD register also enables and masks SERR signals.
Logging Details	FSB does not capture error logging information beyond the event flags in the FSB_FERR, FSB_NERR and PCISTS
Reporting Multiple Errors	The FSB_NERR register captures “next” errors. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	FSB passes along error information to poison data.

Although the IMCH supports all of the errors Table 5-4 lists with the features in Table 5-5, the EP80579 implementation does not take full advantage of these capabilities since its FSB implementation does not support all of the features necessary.

For additional discussion on the IMCH responses to transactions from the FSB interface, see Section 10.1, “Overview”.

5.3.6 Unit-Level Errors from the NSI

The IMCH hardware captures error events from the NSI interface that connects the IMCH to the IICH in the NSI_FERR and NSI_NERR registers. The NSI interface reports an error event to the IA-32 core through IA SCI, SMI, SERR, or MCERR signals based on the settings in the NSI_SCICMD, NSI_SMICMD, NSI_SERRCMD, and NSI_MCERRCMD registers. Software can independently configure the specific signal that each buffer unit error event uses.

Table 5-6 summarizes the error conditions that the NSI can generate.

Table 5-6. Summary of IMCH NSI Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Unsupported Request	Uncorrectable	Fatal, Non-Fatal	SCI, MCERR, SMI, or SERR	Unsupported request detected.
Malformed TLP	Uncorrectable	Fatal, Non-Fatal	SCI, MCERR, SMI, or SERR	Malformed TLP detected.
Receiver Overflow	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Overflow detected in posted, non-posted, or completion upstream queue.
Unexpected Completion	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Completion received that does not correspond to an outstanding request.
Completer Abort	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Completer abort detected.
Completion Timeout	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Request not completed within timeout window.
Poisoned TLP	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Portion of TLP data payload was corrupt.
Data Link Protocol Error	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Error detected in data link protocol.
Replay Timer Timeout	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Replay timer expired.
REPLAY_NUM Rollover	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Retry buffer replay counter rolled over.
Bad DLLP CRC	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Calculated DLLP CRC did not equal received value.
Bad TLP CRC	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Calculated TLP CRC did not equal received value.
Receiver Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Packet framing error.
Received Fatal Error Message	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Fatal error message received over NSI link.
Received Non-Fatal Error Message	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Non-fatal error message received over NSI link.

Table 5-6. Summary of IMCH NSI Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Received Correctable Error Message	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Correctable error message received over NSI link.
Parity Error on Data from Core	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error detected on data received from core.
Link Down	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Link transitioned from DL_UP to DL_DOWN.

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on NSI_SCICMD, NSI_SMICMD, NSI_SERRCMD, and NSI_MCERRCMD register values.

Table 5-7 summarizes the capabilities of the NSI error handling for each of the features that the unit is expected to provide.

Table 5-7. Summary of IMCH NSI Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The NSI_EMASK, NSI_SCICMD, NSI_SMICMD, NSI_SERRCMD, and NSI_MCERRCMD registers enables and masks error reporting. The PCICMD register also enables and masks SERR signals.
Logging Details	NSI captures error logging information in the following registers: • All errors: NSI_FERR, NSI_NERR, and PCISTS capture event flags. • Poisoned TLP: PCISTS captures event flags. • Received fatal/non-fatal/correctable error messages: NSI_ERRSID. All of the logging information that the NSI captures relates to the “first” error with the exception of NSI_NERR.
Reporting Multiple Errors	The NSI_NERR register captures “next” errors. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	NSI passes along error information to poison data.

For additional discussion on the IMCH responses to transactions from the NSI interface, see Section 10.1, “Overview”.

5.3.7 Unit-Level Errors from the EDMA Engine

The IMCH EDMA unit captures error events from the EDMA engine in the EDMA_FERR and EDMA_NERR registers (The EDMA unit reports an error event to the IA-32 core through IA SCI, SMI, SERR, or MCERR signals based on the settings in the EDMA_SCICMD, EDMA_SMICMD, EDMA_SERRCMD, and EDMA_MCERRCMD registers. Software can independently configure the specific signal that each EDMA unit error event uses.

Table 5-8 summarizes the error conditions that the EDMA can generate.

Table 5-8. Summary of IMCH EDMA Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
NDAR Addressing Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Descriptor pointer is of incorrect type or range for channels 0-3.
NDAR Alignment Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Descriptor pointer is not aligned to an 8 DW boundary for channels 0-3.
Source Address Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Source address does not comply with source type or range for channels 0-3.

Table 5-8. Summary of IMCH EDMA Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Destination Address Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Destination address does not comply with destination type or range for channels 0-3.
Parity Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error during read of source data from system memory for channels 0-3.
Write Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Write to RO descriptor registers when DMA in normal mode for channels 0-3.

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on EDMA_SCICMD, EDMA_SMICMD, EDMA_SERRCMD, and EDMA_MCERRCMD register values.

Table 5-9 summarizes the capabilities of the EDMA error handling for each of the features that the unit is expected to provide.

Table 5-9. Summary of IMCH EDMA Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The EDMA_EMASK, EDMA_SCICMD, EDMA_SMICMD, EDMA_SERRCMD, and EDMA_MCERRCMD registers enables and masks error reporting. The PCICMD register also enables and masks SERR signals.
Logging Details	EDMA does not capture error logging information beyond the event flags in the EDMA_FERR, EDMA_NERR and PCISTS.
Reporting Multiple Errors	The EDMA_NERR register captures “next” errors. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	EDMA passes along error information to poison data.

For additional discussion on the IMCH responses to transactions from the EDMA engine, see Section 10.1, “Overview” and Section 10.2, “IMCH Responses to EDMA Transactions”.

5.3.8 Unit-Level Errors from PCI Express* Ports A0 and A1

The IMCH PCI Express* Port Controllers capture error events from the port A0 and A1 controllers in per-port registers. The register set provides two parallel error reporting mechanisms, one that reports standard errors defined by the PCI Express* specification and a second that reports errors that are specific to the EP80579 PCI Express* implementation (and thus, outside of the standard PCI Express* errors).

The PCI Express* controllers capture errors required by the PCI Express* base specification in the UNCERRSTS and CORERRSTS registers. The controllers capture EP80579-specific errors in the PEAUNITERR register. The errors from both sets of errors are aggregated in the PEAFERR and PEANERR registers. The PCI Express* controllers report an error event to the IA-32 core through IA SCI, SMI, SERR, or MCERR signals based on the settings in the PEAERRDOCMD register.

Table 5-10 summarizes the error conditions that the PCI-Express can generate.

Table 5-10. Summary of IMCH PCI-Express Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Unsupported Request	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Request type unsupported.
ECRC Error	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Error in ECRC.
Malformed TLP	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Malformed TLP, like it says...
Receiver Overflow	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Overflow on upstream queue.
Unexpected Completion	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Received completion that does not match any outstanding requests.
Completer Abort	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Received request violates programming model.
Completion Timeout	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	Timeout.
Flow Control Protocol Error	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	
Poisoned TLP	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	TLP data payload corrupt.
Data Link Protocol	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	ACK/NACK has incorrect sequence number.
Unsupported Request	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	
Unsupported Request	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	
Unsupported Request	Uncorrectable	UNCERRSEV[c]	SCI, MCERR, SMI, or SERR	
Replay Timer	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Replay timer expired.
REPLAY_NUM Rollover	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Retry buffer counter rolled over.
Bad DLLP Status	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Computed DLLP CRC does not match received value.
Bad TLP status	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Computed TLP CRC does not match received value.
Receiver Error	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Received 8b/10b error.
LLE Protocol Error	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Transaction layer detected protocol error.
Link Down Error	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Link transitions from DL_UP to DL_DOWN.
Downstream Data Queue Parity	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Parity error occurred in downstream data queue.
SMB Clock Timeout	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	SMB CLK low greater than 25ms.
Unexpected NAK on SMB	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	Unexpected NAK on SMB.
SMB Arbitration	Uncorrectable	Non-Fatal	SCI, MCERR, SMI, or SERR	SMB lost bus arbitration.

a. Fatal versus non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on PEAERRDOCMD register value.
c. Severity is set by the UNCERRSEV register.

Table 5-11 summarizes the capabilities of the PCI-Express error handling for each of the features that the unit is expected to provide.

Table 5-11. Summary of IMCH PCI-Express Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The UNCERRMSK, UNCEDMASK, CORERRMSK, COREDMASK, RPMERRSTS, PEAMASKERR, RPERRCMD, and PEAERRDOCMD registers enables and masks error reporting. The PCICMD register also enables and masks SERR signals.
Logging Details	PCI Express* controllers captures error logging information in the following registers: • RPMERRSTS errors: ERRSID captures requester IDs. • Errors with Header Capture: HDRLOG0, HDRLOG1, HDRLOG2, and HDRLOG3 captures the first four 32-bit words of the headers. This information is in addition to the status information in UNCERRSTS, CORERRSTS, RPMERRSTS, PEAFERR, and PEANERR.
Reporting Multiple Errors	The PEANERR and RPERRMSTS registers captures “next” errors. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	PCI Express* controllers pass along error information to poison data.

See Section 16.4, “PCI Express* Port A Standard and Enhanced Registers: Bus 0, Devices 2 and 3, Function 0” for additional details.

For additional discussion on the IMCH responses to transactions from the PCI Express* ports, see Section 10.1, “Overview”.

5.4 Error Reporting by the IICH

The EP80579 IICH devices rely on the PCI error reporting architecture to reporting errors. In this architecture, details on the errors are logged in the PCI status register from the per-device PCI configuration header. A parity or other severe system error causes the device to generate an IA SERR signal.

Errors that occur on the NSI bus between the IMCH and IICH report through the NSI_FERR and NSI_NERR infrastructure as Section 5.3.1, “Overview of the First and Next Error Architecture” on page 141 and Section 5.3.6, “Unit-Level Errors from the NSI” on page 145 describe.

On the IICH backbone, IICH devices must rely entirely on the SERR signal that the device generates on an error event to report the error. The IICH does not provide any other capability (such as data poisoning) that would allow a consumer of IICH data to note an error. As a result, an IICH device can return erroneous data on a request.

The remainder of this section describes the error handling capabilities of the units in the IICH.

5.4.1 SMBus Interface

The IICH provides a SMBus controller that can generate an interrupt or SMI on error events and can also use the PCI SERR infrastructure to report errors. The HCFG register selects either SMI or interrupt signaling; parity and system errors always signal through SERR. Table 5-12 summarizes the error conditions that the controller reports.

Table 5-12. Summary of SMBus Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Device Error	Uncorrectable	Fatal	Interrupt, SMI[b]	Device error.
Bus Error	Uncorrectable	Fatal	Interrupt, SMI[b]	Bus error.
Failed Bus Transaction	Uncorrectable	Fatal	Interrupt, SMI[b]	Bus transaction failed.
Parity Error	Uncorrectable	Fatal	SERR	Parity error detected.
System Error	Uncorrectable	Fatal	SERR	System error detected.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.
b. Based on HCFG register values.

Table 5-13 summarizes the capabilities of the SMBus controller error handling for each of the features that the unit is expected to provide.

Table 5-13. Summary of SMBus Controller Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CMD and USBINTR registers enables and masks error reporting.
Logging Details	The USB 1.1 interface captures the type of event detected in the DSR, HSTS, and AUXS registers.
Reporting Multiple Errors	The SMBus interface does not capture multiple events.
Data Poisoning	IICH backbone does not support data poisoning.

For additional details on error handling in the SMBus controller, see Section 24.0, "SMBus Controller Functional Description: Bus 0, Device 31, Function 3".

5.4.2 LPC Interface

The IICH provides a LPC interface that uses the PCI SERR infrastructure to report errors. Table 5-14 summarizes the error conditions that the controller reports.

Table 5-14. Summary of LPC Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Parity Error	Uncorrectable	Fatal	SERR	Parity error detected.
System Error	Uncorrectable	Fatal	SERR	System error detected.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-15 summarizes the capabilities of the LPC interface error handling for each of the features that the unit is expected to provide.

Table 5-15. Summary of LPC Interface Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CMD register supports error enabling and masking.

Table 5-15. Summary of LPC Interface Error Reporting Capabilities

Feature	Implementation
Logging Details	The LPC interface captures the type of event detected in the STS register.
Reporting Multiple Errors	The LPC interface does not capture multiple events.
Data Poisoning	IICH backbone does not support data poisoning.

For additional details on error handling in the LPC interface, see Section 19.0, "LPC Interface: Bus 0, Device 31, Function 0".

5.4.3 USB 1.1 Interface

The IICH provides a USB 1.1 controller that can generate an interrupt on error events and can also use the PCI SERR infrastructure to report errors. Table 5-16 summarizes the error conditions that the controller reports.

Table 5-16. Summary of USB 1.1 Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Host Controller Process Error	Uncorrectable	Fatal	Interrupt	Consistency check by host controller fails while processing a TD.
Host System Error	Uncorrectable	Fatal	Interrupt	Serious error during host system access involving HC module.
USB Error	Uncorrectable	Fatal	Interrupt	USB transaction completion ended in error.
Parity Error	Uncorrectable	Fatal	SERR	Parity error on read completion returned to host controller or UHCI register write.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-17 summarizes the capabilities of the USB 1.1 controller error handling for each of the features that the unit is expected to provide.

Table 5-17. Summary of USB 1.1 Interface Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CMD and USBINTR registers supports error enabling and masking.
Logging Details	The USB 1.1 interface captures the type of event detected in the DSR and USBSTS registers.
Reporting Multiple Errors	The USB 1.1 interface does not capture multiple events, error events cause the interface to halt operation until serviced by software.
Data Poisoning	IICH backbone does not support data poisoning.

For additional details on error handling in the USB 1.1 controller, see Section 25.0, "USB (1.1) Controller: Bus 0, Device 29, Function 0".

5.4.4 USB 2.0 Interface

The IICH provides a USB 2.0 controller that can generate an interrupt on error events and can also use the PCI SERR infrastructure to report errors. Table 5-18 summarizes the error conditions that the controller reports.

Table 5-18. Summary of USB 2.0 Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Host System Error	Uncorrectable	Fatal	Interrupt	Serious error during host system access involving HC module.
USB Error	Uncorrectable	Fatal	Interrupt	USB transaction completion ended in error.
Parity Error	Uncorrectable	Fatal	SERR	Parity error on USB read completion.
System Error	Uncorrectable	Fatal	SERR	Parity error on address, command, or data, or unsuccessful completion of ECH-initiated read.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-19 summarizes the capabilities of the USB 2.0 controller error handling for each of the features that the unit is expected to provide.

Table 5-19. Summary of USB 2.0 Interface Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CMD and USB20INTR registers supports error enabling and masking.
Logging Details	The USB 2.0 interface captures the type of event detected in the DSR and USB20STS registers.
Reporting Multiple Errors	The USB 2.0 interface does not capture multiple events, error events cause the interface to halt operation until serviced by software.
Data Poisoning	IICH backbone does not support data poisoning.

For additional details on error handling in the USB 2.0 controller, see Section 26.0, "USB 2.0 Host Controller: Bus 0, Device 29, Function 7".

5.4.5 SATA Interface

The IICH provides a SATA interface that can generate an interrupt on error events and can also use the PCI PERR infrastructure to report errors. Table 5-20 summarizes the error conditions that the controller reports.

Table 5-20. Summary of SATA Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Host Bus Fatal Error	Uncorrectable	Fatal	Interrupt	Unrecoverable host bus error
Host Bus Data Error	Uncorrectable	Fatal	Interrupt	Uncorrectable data error.
Interface Fatal Error	Uncorrectable	Fatal	Interrupt	Fatal error on SATA interface.
Interface Non-Fatal Error	Uncorrectable	Non-Fatal	Interrupt	Non-fatal error on SATA interface.
Parity Error	Uncorrectable	Fatal	SERR	Parity error detected on interface.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-21 summarizes the capabilities of the SATA controller error handling for each of the features that the unit is expected to provide.

Table 5-21. Summary of SATA Interface Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CMD and PIE[0-3] registers support error enabling and masking.
Logging Details	The SATA interface captures the type of event detected in the STS and PIS[0-3] registers.
Reporting Multiple Errors	The SATA interface does not capture multiple events.
Data Poisoning	IICH backbone does not support data poisoning.

For additional details on error handling in the SATA interface, see Section 23.0, "SATA: Bus 0, Device 31, Function 2".

5.4.6 Serial I/O Interface

The IICH provides a serial I/O interface that can generate an interrupt on error events. Table 5-22 summarizes the error conditions that the serial I/O interface captures.

Table 5-22. Summary of Serial I/O Interface Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Framing Error	Uncorrectable	Fatal	Interrupt	Received character missing stop bit.
Parity Error	Uncorrectable	Fatal	Interrupt	Received character has parity error.
Overrun Error	Uncorrectable	Fatal	Interrupt	Receive buffer over-written.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-23 summarizes the capabilities of the thermal sensor error handling for each of the features that the unit is expected to provide.

Table 5-23. Summary of Serial I/O Interface Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The IER and LCR registers supports error enabling and masking.
Logging Details	The serial I/O interface captures the type of event detected in the IIR and LSR registers.
Reporting Multiple Errors	The serial I/O interface does not capture multiple events.
Data Poisoning	N/A

For additional details on error handling in the serial I/O interface, see Section 33.0, "Serial I/O Unit and Watchdog Timer".

5.5 Error Reporting by the System Memory Controller

The memory controller interfaces with memory to provide data movement to and from DRAM along the AIOC-direct and coherent paths to memory. The memory controller is designed to conform to the IMCH first and next error handling architecture that Section 5.3.1, "Overview of the First and Next Error Architecture" on page 141 describes. The memory controller reports its unit-level first/next error events through the DRAM_FERR and DRAM_NERR registers in the memory controller (see Section 11.5, "Error Handling"). The flow of memory controller errors matches the IMCH behavior for its other errors as Section 5.3.1 describes.

5.5.1 Handling Out-of-Bounds Addresses

The memory controller does not perform any bounds checking on system addresses that it receives from other agents. As a result, the behavior of the EP80579 in response to an access to a DRAM location that is outside the range of populated memory depends on the source of the transaction and the path it takes through the device. There are two paths of relevance to this discussion: the Coherent path through the IMCH and the AIOC-Direct path directly to the memory controller.

Accesses to locations above top of memory along the Coherent path (from either AIOC agents via the AIOC-interface, the IA-32 core, or other agents attached to the CMI through PCI Express*, etc.) will be aborted by the IMCH. The specific manner of the abort depends on whether the transaction originates from the IA-32 core or a AIOC agent.

Access to locations above top of memory along the AIOC-Direct path (from the AIOC Memory Target) are not aborted or otherwise trapped by default. The results of such transactions are undefined and may alias onto populated memory regions. As a result, software must program the EP80579 so that it can never generate an AIOC-Direct system address above TOM. The Memory Target provides the ability to log range errors (and optionally halt the AIOC master that generated the transaction).

5.5.2 IMCH - Memory Controller

The memory controller can notify the IA-32 core of memory-related error events through SCI, SMI, SERR, or MCERR signals based on the settings in the DRAM_SCICMD, DRAM_SMICMD, DRAM_SERRCMD, and DRAM_MCERRCMD registers (see Section 11.0, “System Memory Controller”). Software can configure the specific signal used for each error event independently.

Table 5-24 summarizes the error conditions that the memory controller captures. Note that the memory controller always reports errors through the non-fatal classification in the error reporting registers.

Table 5-24. Summary of Memory Controller Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Uncorrectable Write Error	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Write of poisoned data to DRAM.
Uncorrectable Read Error	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Error during normal demand reads.
Uncorrectable Scrubber Data Error	Uncorrectable	Fatal	SCI, MCERR, SMI, or SERR	Memory scrubber encountered an uncorrectable error.
Correctable Read Error	Correctable	Non-Fatal	SCI, MCERR, SMI, or SERR	Hardware will correct and report if appropriately configured
Error Threshold Detect	Status	Non-Fatal	SCI, MCERR, SMI, or SERR	Count of single- or double-bit errors exceeds a programmable threshold.
Memory Test Complete	Status	Non-Fatal	SCI, MCERR, SMI, or SERR	Status event to indicate when memory test hardware completes testing.

a. “Fatal” events result in data loss or data corruption that the unit cannot repair, “Non-Fatal” events do not. This can differ from the fatal/non-fatal classification for reporting through GLOBAL_FERR and GLOBAL_NERR.
b. Based on DRAM_SCICMD, DRAM_SMICMD, DRAM_SERRCMD, and DRAM_MCERRCMD register values.

Table 5-25 summarizes the capabilities of the memory controller error handling for each of the features that the unit is expected to provide.

Table 5-25. Summary of Memory Controller Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The DRAM_EMASK, DRAM_SCICMD, DRAM_SMICMD, DRAM_SERRCMD, and DRAM_MCERRCMD registers enable and mask error reporting.
Logging Details	Memory Controller captures additional error logging information in the following registers: • Uncorrectable read errors: DRAM_DED_ADD. • Uncorrectable scrubber data errors: DRAM_SCRB_ADD. • Correctable read errors: DRAM_SECF_ADD, DRAM_SECF_SYNDROME, DRAM_SECN_ADD, DRAM_SECN_SYNDROME. • Error threshold detect: RANKTHREX, THRESH_SEC0, THRESH_SEC1, THRESH_DED, DRAM_SEC_R0, DRAM_SEC_R1, DRAM_DED_R0, DRAM_DED_R1. Additional logging information is not captured for the remaining errors in Table 5-24. With the exception of DRAM_SECN_ADD and DRAM_SECN_SYNDROME, all of the logging information that the memory controller captures relates to the “first” error.
Reporting Multiple Errors	The DRAM_NERR register captures the “next” errors seen by the memory controller. This register indicates up to one additional error (beyond the first error) of each type.
Data Poisoning	Memory Controller passes along error information to poison data both on inbound (from memory) data and outbound (to memory) data.

For additional details on error handling in the memory controller, see Section 11.5, “Error Handling”.

5.6 Error Reporting by AIOC Devices

The AIOC devices continue to use the native error reporting infrastructure that each unit provides. Primarily, this infrastructure relies on one or more side-band error signals from each AIOC device that can signal an error along with parity protection on key interfaces. The AIOC native error reporting mechanisms are then bridged into the PCI framework that the EP80579 uses to expose AIOC devices to IA. Although AIOC devices present a PCI interface to IA, they do not implement PCI error reporting capabilities such as SERR. As a result, the native signals from the AIOC devices are bridged onto PCI INTx or MSI signals. This implies that the driver software provides all error handling for AIOC units[1].

The following sections describe the error reporting for each of the AIOC units along with transaction responses. The per-unit presentations are organized by the PCI device in which the units materialize.

5.6.1 Gigabit Ethernet MAC

The Gigabit Ethernet MAC units each signal error conditions through three interrupt signals: Functional 0, Functional 1, and Error. Software uses the IMS0, IMS1, and IMS2 configuration registers in a Gigabit Ethernet MAC to map error conditions onto the Functional 0, Functional 1, and Error interrupt signals, respectively. Depending on the configuration, error and functional events may share an interrupt (e.g., software may configure the functional 1 interrupt to signal both error and functional events); typically, software will configure a MAC to deliver its error events separately through only the error interrupt.

Table 5-26 summarizes the error conditions that the Gigabit Ethernet MAC captures.

1. If PCI abstractions such as SERR were used, this would not be the case. Platform and/or O/S software would also be involved in error handling for the devices even if the involvement is limited to generating a blue screen.

Table 5-26. Summary of Gigabit Ethernet MAC Error Conditions

Event	Type	Fatality[a]	Reports via[b]	Notes
Statistic Register ECC Error	Uncorrectable	Fatal	ERR, FN0, or FN1 Interrupts	Double-bit ECC error in a statistic register.
Internal Memory Error	Uncorrectable	Fatal	ERR, FN0, or FN1 Interrupts	Parity or double-bit ECC error in an internal memory.
DMA Packet Buffer Error	Uncorrectable	Fatal	ERR, FN0, or FN1 Interrupts	Double-bit ECC error during read from DMA packet buffer on Tx or Rx.
DMA Tx Desc. ECC Error	Uncorrectable	Fatal	ERR, FN0, or FN1 Interrupts	Double-bit ECC error during read from DMA transmit descriptor.
DMA Rx Desc. ECC Error	Uncorrectable	Fatal	ERR, FN0, or FN1 Interrupts	Double-bit ECC error during read from DAM receive descriptor.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.
b. Based on settings in IMS0, IMS1, and IMS2 registers.

Table 5-27 summarizes the capabilities of the Gigabit Ethernet MAC error handling for each of the features that the unit is expected to provide.

Table 5-27. Summary of Gigabit Ethernet MAC Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The IMS0, IMS1, and IMS2 interrupt mask set registers and the IMC0, IMC1, and IMC2 interrupt mask clear registers support error enabling and masking. When software configures the GbE to deliver errors on their own interrupt, the SMIA and SMME registers from the signal target capability in the PCI configuration header for a GbE MAC can also support error enabling and masking.
Logging Details	The SINT register from the signal target capability in the PCI configuration header for a GbE MAC provides read-only access to the state of the interrupt signals from a GbE MAC. Additional logging information is not captured for the other errors in Table 5-26.
Reporting Multiple Errors	Individual status bits in the GbE ICR0, ICR1, and ICR2 interrupt cause registers are set as conditions occur. The unit can indicate at most one outstanding error at any time.
Data Poisoning	GbE passes along error information to poison data.

See Section 37.6, "GbE Controller Register Summary" and Section 37.5.12, "Error Handling" for additional details.

5.6.2 CAN Interface

The CAN units signal error conditions through two interrupt signals. The CAN unit shares one interrupt between functional signaling duties (e.g., signaling that a message was received) and error reporting, while the second interrupt reports only parity errors. Status and enable registers in the CAN operate and control signaling functionality in the CAN such as error reporting.

Table 5-28 summarizes the error conditions that the CAN captures.

Table 5-28. Summary of CAN Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
CRC Check Error	Uncorrectable	Fatal	CAN System Interrupt	Mismatch between received and computed CRC.
Acknowledge Error	Uncorrectable	Fatal	CAN System Interrupt	Improperly formatted ACK slot.
Form Error	Uncorrectable	Fatal	CAN System Interrupt	Improperly formatted fixed-form field.
Bit Error	Uncorrectable	Fatal	CAN System Interrupt	Mismatch between monitored and sent bit value.
Stuff Error	Uncorrectable	Fatal	CAN System Interrupt	Improperly formatted message from Start of Frame to CRC delimiter.
CAN SRAM Parity Error	Uncorrectable	Fatal	CAN Parity Interrupt	Parity error in interface SRAM. Clearing the parity error requires a reset of the CAN interface. Please see section on CAN interrupts in Section 39.2, "Feature List" on page 1569 for more details.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-29 summarizes the capabilities of the CAN error handling for each of the features that the unit is expected to provide.

Table 5-29. Summary of CAN Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	The CAN interrupt enable register supports error enabling and masking. The SMIA and SMME registers from the signal target capability in the PCI configuration header for the CAN units also supports error enabling and masking. Using these registers to mask the CAN System Interrupt masks both error and functional events since the CAN units use this interrupt to signal both error and functional conditions.
Logging Details	The SINT register from the signal target capability in the PCI configuration header for a CAN unit provides read-only access to the state of the interrupt signals from a CAN unit. CAN does not log additional details on errors.
Reporting Multiple Errors	Individual status bits in CAN interrupt status register are set as conditions occur. The unit can indicate at most one outstanding error of each type at any time.
Data Poisoning	CAN does not require support for data poisoning. Errors during transactions cause the transaction to abort and an error event to be signaled.

See Section 39.6, "Register Summary" and Section 39.5.2, "Error Handling" for additional details.

5.6.3 SSP Interface

The SSP unit signals error conditions through a single interrupt signal. The SSP block shares this interrupt between functional duties (e.g., transmit FIFO service request) and error reporting duties. Software is expected to use status registers in the SSP to determine the cause of a signal.

Table 5-30 summarizes the error condition that the SSP captures.

Table 5-30. Summary of SSP Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Functional Receiver Overrun (ROR)	Uncorrectable	Fatal	SSP Interrupt	Receive FIFO is full, any incoming data is discarded.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-31 summarizes the capabilities of the SSP error handling for each of the features that the unit is expected to provide.

Table 5-31. Summary of SSP Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	SSP does not provide the ability to enable or mask the interrupt from an ROR error condition. The SMIA and SMME registers from the signal target capability in the PCI configuration header for the SSP unit also support error enabling and masking. Using these registers to mask the SSP Interrupt masks both error and functional events since the SSP unit uses this interrupt to signal both error and functional conditions.
Logging Details	The SINT register from the signal target capability in the PCI configuration header for the SSP unit provides read-only access to the state of the interrupt signals from SSP. SSP does not log additional details on its errors.
Reporting Multiple Errors	SSP can only generate a single error. The unit can indicate at most one outstanding error at any time.
Data Poisoning	SSP error conditions result in loss of data and system interrupt. There is no need to poison in these cases.

See Section 40.4, "Register Summary"and Section 40.3.2, "Error Handling" for additional details.

5.6.4 Local Expansion Bus

The Local Expansion Bus unit signals error conditions from the interface into the EP80579 through a single interrupt signal that is used exclusively for errors. In addition, this unit signals errors from the internal bridge through a separate interrupt.

Table 5-32 summarizes the error conditions that the local expansion bus captures.

Table 5-32. Summary of Local Expansion Bus Error Conditions

Event	Type	Fatality[a]	Reports via	Notes
Parity Error	Uncorrectable	Fatal	LEB Parity Error Interrupt	Parity error on outbound read from the EP80579.

a. "Fatal" events result in data loss or data corruption that the unit cannot repair, "Non-Fatal" events do not.

Table 5-33 summarizes the capabilities of the local expansion bus error handling for each of the features that the unit is expected to provide.

Table 5-33. Summary of Local Expansion Bus Error Reporting Capabilities

Feature	Implementation
Enabling and Masking Error Reporting	LEB device provides the following registers to support error enabling and masking: • Errors from LEB: EXP_TIMING_CS[0-7]. The SMIA and SMME registers from the signal target capability in the PCI configuration header for the LEB also support error masking and enabling.
Logging Details	The SINT register from the signal target capability in the PCI configuration header for the LEB provides read-only access to the state of the interrupt signals from the LEB. LEB device captures additional error logging information in the following registers: • LEB parity errors: EXP_PARITY_STATUS.
Reporting Multiple Errors	LEB device reports additional errors as follows: • LEB errors
Data Poisoning	LEB pass along error information to poison data.

See Section 42.5, “Register Summary”.

5.6.5 IEEE 1588, and GCU

The IEEE 1588 and GCU in the AIOC do not signal any non-functional error conditions.

§ §

6.0 Reset and Power Management

This chapter describes the Intel® EP80579 Integrated Processor reset and power management.

6.1 Reset and Powergood Distribution

This section discusses the detailed sequencing of how power and clocking signals must be applied to bring the EP80579 out of reset.

6.1.1 Types of Reset

The EP80579 has four types of reset: Power-good (cold) reset, Hard reset, CPU-only reset, and Targeted I/O subsystem reset(s). Each of these reset subclasses have unique effects and is described in Table 6-1.

Table 6-1. Types of Reset and Wake-up from Power Saving States

Type	Mechanism	Effect of Reset on following blocks				
		CPU	IMCH/ IICH	AIOC	PCI-E	DDR
Power-good	Input pin	Reset	Reset	Reset	Reset	Reset
Hard	Input pin	Reset	Reset	Reset	Reset	Reset
Software (SW) Controlled	Write to I/O port CF9	Reset	Reset	Reset	Reset	Reset
CPU-only	Internal to the EP80579. Generated by IMCH	Reset	-N/A	-N/A	-N/A	-N/A
S3 -> S0	Wake Event	Reset	Reset	Reset	Reset	-N/A
S4/S5->S0	Wake Event	Reset	Reset	Reset	Reset	Reset

6.1.1.1 Powergood Implementation

The initial boot from when the power supplies are energized is facilitated by the Powergood mechanism. The voltage sources from all platform power supplies are routed to a system component which tracks them as they ramp-up, asserting platform powergood (CPU_VRD_PWR_GD and SYS_PWR_OK) after a fixed interval (nominally 99 ms) after the last voltage reference has stabilized. Powergood signals are propagated asynchronously to dedicated IICH, IMCH, and IA-32 core inputs. Generally, the devices on AIOC fabric including the AIOC reset block do not receive a powergood signal. The exception is the GbE MAC devices (see Section 6.1.2.3.5, “GbE MAC”).

Table 6-2 summarizes the power wells and external voltages required by the EP80579. More detailed power supply pin information can be found in Table 6-3.

Table 6-2. Power Wells and External Voltages

Power Well	Nominal Voltage	Components
Core	1.0-1.3 V	IA Processor: IA-CPU
	1.2 V	Core Logic (IMCH, IICH, ASU, SSU, TDM, GbE MAC1, GbE MAC2), SATA pads, PCI-E pads, Local Expansion Bus
	1.8 V	PCI-E PLL, DDR2 pads. Note: 0.9V are generated from 1.8V
	2.5 V	RMII/RGMII
	3.3 V	SATA, PCI-E
	5.0V	5V tolerance reference
Suspend	1.2 V	IICH, USB pads, DDR2 core logic, GbE MAC0, USB core logic and pads
	2.5 V	RMII/RGMII
	3.3 V	USB pads, RMII/RGMII pads
	5.0V	5V sustain reference
RTC	3.3 V	RTC

6.1.1.2 Hard Reset Implementation

A hard reset is initiated by the IICH via the PLTRST# as a result various S-state wake events or other reset signal assertions. PLTRST# is driven to the IMCH and is propagated from there to the reset block for AIOC fabric. The reset block in the AIOC fabric is responsible for resetting the individual blocks.

IMCH propagates a hard reset to the FSB and subordinate PCI Express* subsystems. The FSB components are reset via the CPURST# (internal signal) signal, while the PCI Express* subsystems through PCIRST#.

6.1.1.3 Software Controlled Reset

Software may cause a full system reset through a write to the Reset Control Register located at I/O port CF9. See also Section 6.1.1.4, “CPU Only Reset Implementation” for software controlled CPU only reset mechanisms.

6.1.1.4 CPU Only Reset Implementation

For power management, error conditions and other reasons, the EP80579 supports a targeted CPU only reset semantic. This mechanism eliminates system reset at large when the CPU function (such as clock gearing selection) must be updated during initialization. It only affects the IA-32 core. Other blocks such as IICH, IMCH, AIOC complex are not reset. It is controlled by IMCH.

The IA-32 core can also be reset via the assertion of its INIT# pin which may be accomplished by several conditions including a software write to the Reset Control Register in IICH. Asserting the INIT# pin on the IA-32 core invokes a response similar to that of asserting CPURST#. The major difference is that during an INIT, the internal caches, MSRs, MTRRs, and FPU state are left unchanged (although, the TLBs and BTB are invalidated as with a hardware reset). When INIT is signaled while the processor is in virtual-8086 mode, the processor leaves virtual-8086 mode and enters real-address mode. An INIT provides a method for switching from protected to real-address mode while maintaining the contents of the internal caches.

6.1.1.5 S-state Wake Events

Wake events for the various ACPI sleep states cause a hard reset to the EP80579 (See Section 27.6.3, “Exiting Sleep States”). Wake on LAN, supported by the GbE MACs, leverages mechanisms provided for GPI and PME wake events.

6.1.1.6 Targeted Reset Implementation

The targeted reset is provided for hot-plug events, as well as for port specific error handling under MCA or SMI software control.

A targeted reset may be requested by setting bit six (Secondary Bus Reset) of the Bridge Control register (D2, F0, offset 3Eh) in the target root port device. Setting this bit crashes the Link Training and Status State machine (LTSSM) of the target port to the reset state, where it issues at least 1024 TS1 ordered sets with the reset bit asserted. This propagates an in-band “hot” reset to the downstream device, and consequently force an equivalent reset to any devices further downstream. This reset is identical to a general hard reset from the perspective of destination PCI Express* device.

6.1.2 Platform Reset and Powergood

This section describes the reset and powergood external platform interfaces.

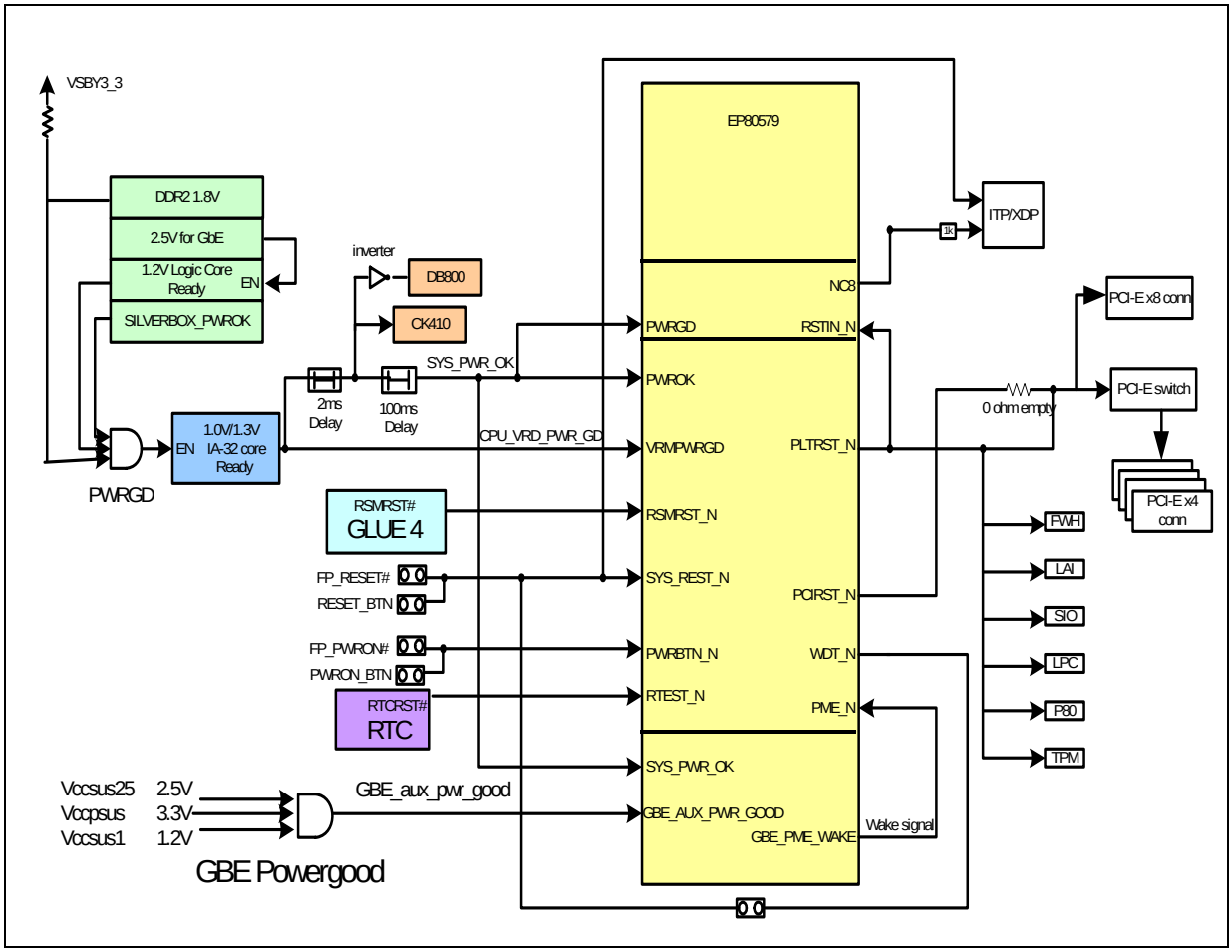
6.1.2.1 Platform Powergood

The EP80579 receives two powergood signals from the platform. The first is CPU_VRD_PWR_GD and the other is SYS_PWR_OK. SYS_PWR_OK is asserted after a fixed delay from the time that CPU_VRD_PWR_GD goes active and indicates that power has been stable for at least 99 ms. The EP80579 inputs PWRGD, PWROK, and SYS_PWR_OK are connected to the SYS_PWR_OK platform signal. CPU_VRD_PWR_GD and SYS_PWR_OK distribution inside the EP80579 is discussed in Section 6.1.2.3, “Reset and Powergood Distribution”. Refer to Figure 6-1 for the block diagram showing CPU_VRD_PWR_GD and SYS_PWR_OK interfaces.

6.1.2.2 Platform Reset

The EP80579 receives two reset signals from the platform. The first one is SYS_RESET which includes the reset button on the platform. The second is Resume Reset which is used for resetting the IICH resume well after power is restored from a power failure. Reset distribution inside the EP80579 is discussed in Section 6.1.2.3, “Reset and Powergood Distribution”. Refer to the Figure 6-1, for a block diagram showing the reset interface.

Figure 6-1. Powergood and Reset Interface



6.1.2.3 Reset and Powergood Distribution

The EP80579 reset follows a general path through the IICH to the IMCH and out to the rest of the chip.

6.1.2.3.1 IICH

IICH plays the central role in reset and powergood distribution to the whole chip. IICH receives two powergood signals from the platform (CPU_VRD_PWR_GD and SYS_PWR_OK). The assertion of these signals starts the reset sequence for the EP80579.

IICH generates the central reset signal (known as PLTRST#) that initiates the reset of the rest of the chip. PLTRST# is received by IMCH. IICH also generates PCIRST# signal for resetting the PCI device. PCIRST# is similar to PLTRST# except that PCIRST# can be asserted using a CSR.

IICH also generates the powergood (CPUPWRGD#) signal for IA-32 core.

IICH Signals

Inputs

VRMPWRGD - Voltage Regulator PowerGood: This signal is directly connected to the platform signal, CPU_VRD_PWR_GD and signifies that the voltage regulator is stable.

PWROK - IICH Power Okay: This signal is directly connected to the platform signal, SYS_PWR_OK. When asserted, PWROK is an indication to the IICH that power has been stable for at least 99 ms and that PCICLK has been stable for at least 1 mS. PWROK can be driven asynchronously. When PWROK is inactive, the IICH asserts PLTRST#.

SYS_RESET# - System Reset: This signal initiates a system reset and causes PLTRST# to go active. SYS_RESET# must be asserted for at least 100ms. System reset cannot occur again until SYS_RESET# has been detected inactive, and the system is back to a full S0 state with PLTRST# inactive. If bit 3 of the Reset Control Register is set then the assertion of SYS_RESET# will result in a full power cycle reset.

RSMRST# - Resume Well Reset: This signal resets the IICH resume power plane logic when power is reapplied after a power failure. If the AFTERG3_EN bit in the General Power Management Configuration 3 Register (D31 F0 Offset A4) is set to 0, IICH transitions the system from G3 (mechanical off) to S0 state and causes the assertion of the PLTRST# output. If AFTERG3_EN is 1, the system will transition to S5 state.

RTEST# - RTC Well Test: This signal is tied to the platform RTCRST# signal. Normally it is held high (to VccRTC), but can be driven low on the tester or motherboard to test the RTC well. RTEST# resets some bits in the RTC well that are otherwise not reset by PLTRST# or RSMRST#. An external RC circuit on the RTCRST# signal creates a time delay such that RTCRST# will go high some time after the battery voltage is valid. The RC time delay must be in the 10-20 ms range. This allows detection when a new battery has been installed. Unless entering a XOR Chain test mode, the RTEST# input must always be high when all other non-RTC power planes are on.

Outputs

CPUPWRGD - CPU PowerGood: This signal is the logical AND of the IICH VRMPWRGD and PWROK input signals. This signal is connected to the processor's powergood input to indicate when the processor power is valid.

PLTRST# - Platform Reset: This signal is asserted by SYS_RESET#, RSMRESET#, or software. The IICH asserts PLTRST# to reset devices on the platform (e.g., SIO, FWH, LAN, IMCH, IDE, TPM, etc.) during power-up (CPU_PWRGD de-asserted) and when software initiates a hard reset sequence through the Reset Control register. The IICH drives PLTRST# active a minimum of 1 ms when initiated through the Reset Control register. The IICH de-asserts PLTRST# a minimum of 1 ms after CPU_PWRGD is driven high.

PCIRST# - PCI Reset: This is the secondary PCI bus reset signal. This signal is asserted a small number of PCI clocks after PLTRST# or can be asserted independently by the Secondary Bus Reset bit.

6.1.2.3.2 IMCH

IMCH plays a crucial role in the reset sequence for IA-32 core. IMCH receives the powergood signal (SYS_PWR_OK) from the platform. The IICH PLTRST# drives the IMCH RSTIN# input. IMCH drives the CPURST# (internal signal) while it is in the reset. IMCH works with IICH to initialize the NSI link between IICH and IMCH. Once the central reset (PLTRST#) is de-asserted, IMCH de-asserts the CPURST# (internal signal).

IMCH can also configure some aspects of the IA-32 core at Power On. IMCH drives these configuration settings before CPURST# (internal signal) assertion based on the contents of its Power-On Configuration Register (D8, F0, offset C0h).

IMCH Signals

Inputs

PWRGD - IMCH PowerGood: This signal is directly connected to the platform signal, SYS_PWR_OK. When asserted, PWRGD is an indication to the IMCH that power has been stable for at least 99 ms. When PWRGD is inactive, the IMCH asserts its CPURST# (internal signal) outputs.

RSTIN# - IMCH Reset: This signal is directly connected to the IICH PLTRST# output. When RSTIN is active, the IMCH asserts its CPURST# (internal signal) output.

6.1.2.3.3 IA-32 core

Early in the cold reset (powergood) sequence, the IA-32 core voltage regulator drives a default voltage to the IA-32 core to read the fuses containing the IA-32 core FSB frequency requirements. The EP80579 then drives its BSEL pin to the appropriate value which is latched by the platform when it is known to be stable. The platform uses the BSEL information to update the IA-32 core voltage regulator to the appropriate operating voltage prior to the assertion of CPU_VRD_PWR_GD. The platform also uses the value of BSEL to drive the clock generator to the correct reference clock (BCLK) frequency.

Reset and configuration for IA-32 core is done by IICH and IMCH. Reset for IA-32 core starts when IMCH asserts CPURST# (internal signal) and IICH asserts CPUPWRGD. The assertion of both of these signals initiates the PLL locking process for the IA-32 core. All the flops and internal states are reset during the reset process. The processor's PLL locks before CPURST# (internal signal) is de-asserted. CPURST# (internal signal) needs to be asserted for at least 1ms and not more than 10ms (processor spec).The IA-32 core receives power-on configuration values on its address pins during CPURST# (internal signal).

6.1.2.3.4 IMCH

The IMCH receives the central reset (PLTRST#) from IICH. Clocks to DIMMs are disconnected till BIOS configures the DIMMs. Memory controller core logic boots at the default frequency as driven by the BSEL pin. Based on the DDR type and frequency memory controller needs to re-lock itself at the DDR frequency once BIOS has read the DDR.

6.1.2.3.5 GbE MAC

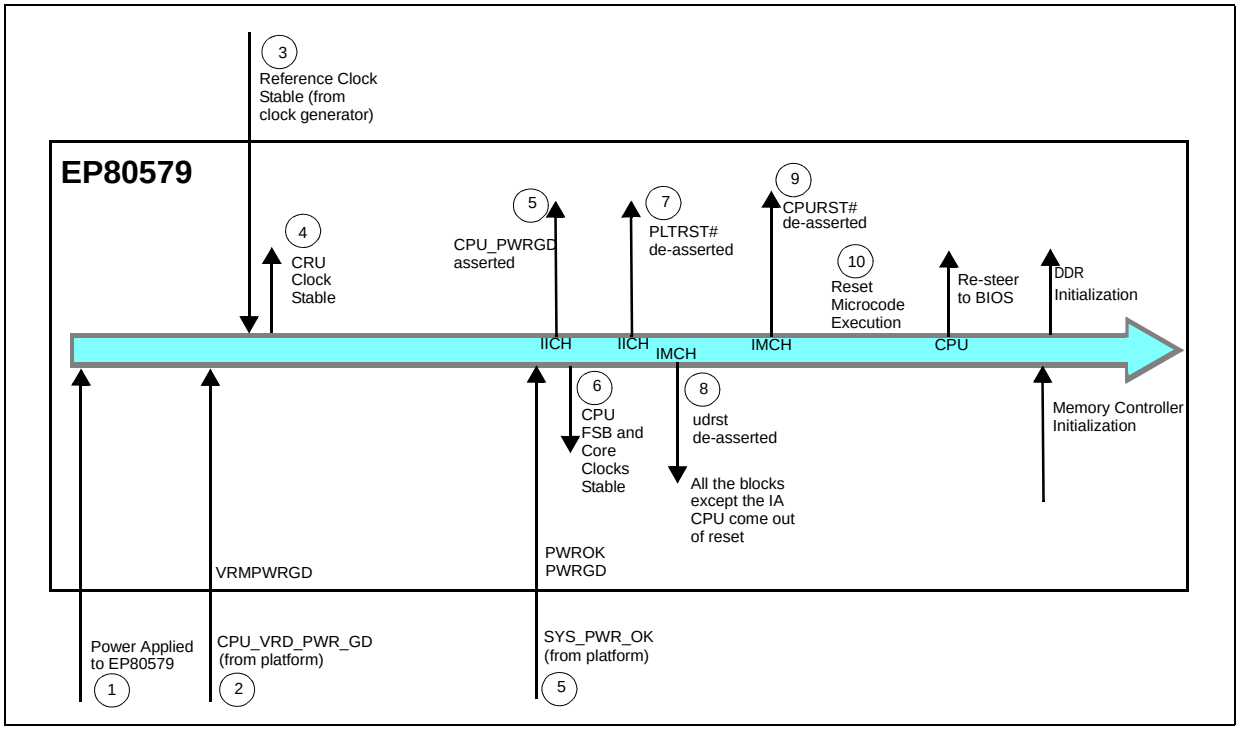
There are three GbE MAC devices. Each GbE receives the internal system reset. Each GbE also receives a power OK signal from the platform that is also used as a reset. GbE0 receives this signal via the GBE_AUX_PWR_GOOD external pin. GbE1 and GbE2 receive this signal via the SYS_PWR_OK external pin. The SYS_PWR_OK pin is connected to the SYS_PWR_OK platform signal which is also connected to the PWROK and PWRGD pins. The GBE_AUX_PWR_GOOD pin should be connected to SYS_PWR_OK when no auxiliary power supply is used. If an auxiliary supply is used for GbE0, then GBE_AUX_PWR_GOOD should be connected to the power good signal from that power supply (this signal is subject to the timing requirements documented in Figure 6-4, "Power Rail Sequence Timings (Sustain Well Power Management)" on page 170).

Under all circumstances, GbE0 MUST be powered by either the system supply or the auxiliary supply. Likewise, GBE_AUX_PWR_GOOD MUST be connected to the corresponding power good signal. GbE0 must be powered to enable operation of either GbE1 or GbE2.

6.1.3 EP80579 Power Sequencing and Reset Sequence

The following diagrams show the reset sequencing.

Figure 6-2. Reset Sequence



1. The EP80579 receives power and drives its BSEL and V_SEL pins. CPU_VRD_PWR_GD, SYS_PWR_OK (platform signals) are not asserted. PLTRST#, and CPURST# (internal signal) are asserted.
2. CPU_VRD_PWR_GD is asserted (Platform signal). Internal signal name is VRMPWRGD.
3. External Reference Clock provided from platform is stable, and is supplied to the EP80579 internal PLLs to generate required internal clocks. Voltage regulator output is modified to correspond to BSEL and V_SEL values.
4. The EP80579 CRU PLL locks.
5. SYS_PWR_OK (platform signal) == PWROK/PWRGD internal signal asserted
6. IO and Core PLLs lock on the CPU.
7. IICH de-asserts PLTRST#
8. IMCH de-asserts udrstb (internal reset unit). All EP80579 blocks except the IA-32 core come out of reset.
9. IMCH de-asserts CPURST# (internal signal) CPU executes the reset micro-code

Table 6-3. EP80579 Power Supply Pins

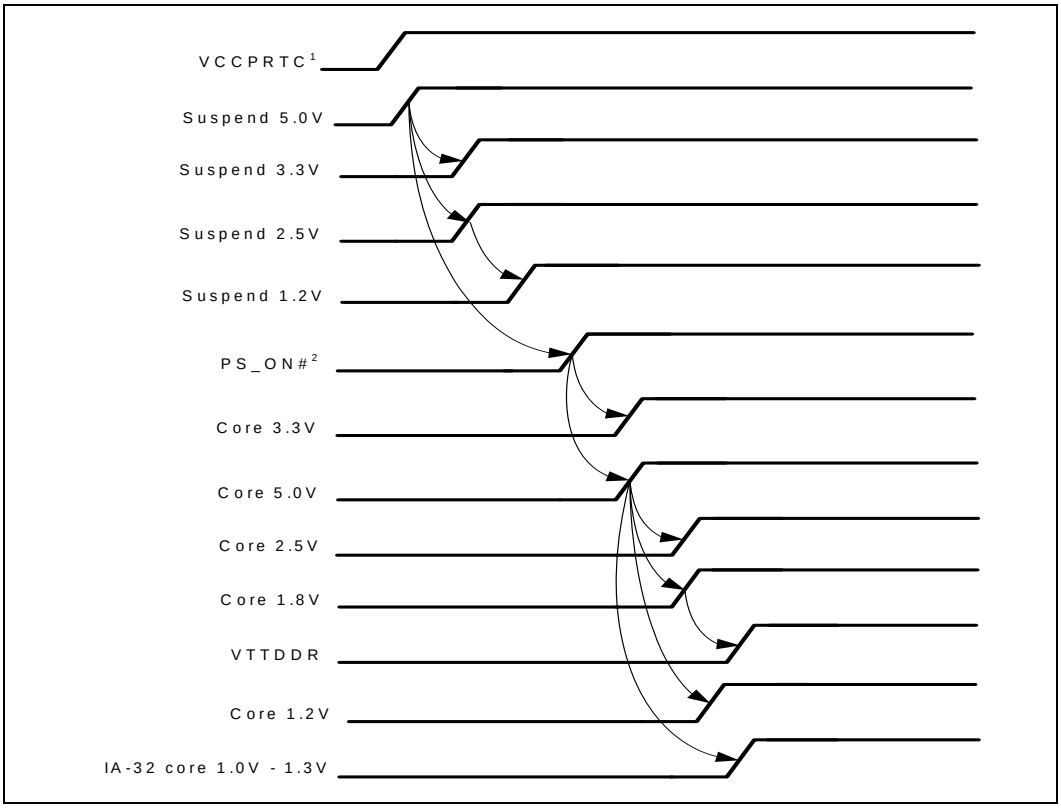
Nominal Voltage	Voltage Tolerance	Package Pin	Supply Types	Description
WELL = CORE				
0.9 V (DDR2)	+/-5%	VTTDDR	DDR	DDR termination voltage
1.0V@600MHz 1.3V@1066/ 1200MHz	+/-2%	VCCVC	IA-32 core	IA-32 core power
1.2 V	+/-5%	VCCA[1]	IA-32 core	IA-32 core PLL1 power
		VCCA[2]	IA-32 core	IA-32 core PLL2 power
		VCC	CRU, CRU_PAD, DDR, Expansion, Bus, GBE, IMCH_PAD, MISC IO, PCI, Express, SATA	Core power
		VCCUSB12	USB2	Digital power
		VCCAUSB12	USB2	Analog power
		VCCAHPLL	CRU	Analog PLL power
		VCCAPE0PLL12	PCI-Express	PLL digital power
		VCCAPLL	SATA	Analog PLL power
		VCCAPE	PCI-Express	Receiver analog power
		VCCAPE	PCI-Express	Receiver analog power
		VCCAPE	PCI-Express	Transmitter analog power
		VCCARX	SATA	Analog receiver power
		VCCATX	SATA	Analog transmitter power
		VCCRPE	PCI-Express	Receiver digital power
		VCCSATA	SATA	SATA power
1.8 V (DDR2)	+/-5%	VCC18	DDR	DDR IO power
		VCC18	TRNG	TRNG power
		VCCTMP18	Thermal Sensor	Thermal sensor power
		VCCAPE0PLL18	PCI-Express	PLL VRM power
2.5 V	+/-5%	VCC25	GBE	GBE IO
3.3 V	+/-5%	VCC33	CRU_PAD, Expansion Bus, IMCH_PAD, MISC IO	IO power
		VCCGBE33	GBE	3.3V tolerance reference
		VCCSATA33	SATA	SATA power
		VCCABG3P3_USB	USB2	Analog bandgap power
		VCCABGP033	PCI-Express	Bandgap analog power
		VCCASATABG3P3	SATA	Analog bandgap power
5 V	+/-5%	VCC50	Expansion Bus, MISC IO, CRU_PAD, IMCH_PAD	5V tolerance reference
WELL = SUSPEND				

Table 6-3. EP80579 Power Supply Pins

Nominal Voltage	Voltage Tolerance	Package Pin	Supply Types	Description
1.2 V	+/-5%	VCC1P2_USBSUS	USB2	1.2V USB sustain power
		VCCSUS1	IMCH_PAD	RTC core sustain power
		VCCSUS1	GBE	Core GBE sustain power
2.5 V	+/-5%	VCCSUS25	GBE	Sustain GBE power
3.3 V	+/-5%	VCCPSUS	USB2, IMCH_PAD	USB and RTC IO sustain power
		VCCGBEPSUS	GBE	Sustain 3.3V tolerance reference
5 V	+/-5%	VCC50_SUS	IMCH_PAD, USB2	5V sustain reference
WELL = RTC				
3.3	+/-5%	VCCPRTC	IMCH_PAD	Real Time Clock power

Figure 6-3 illustrates the sequence that power rails should follow as they are brought up at Power On.

Figure 6-3. EP80579 Rail Power On Sequence



Figures 6-4, 6-5, 6-6, and 6-7 accompanied by Table 6-4, Table 6-5, and Table 6-6 show the relationship between the power supply rails and key reset signals upon EP80579 power-up. In Figures 6-4 and 6-5, the terms “Core” and “Suspend” refer to the power wells that Table 6-2 on page 162 describes.

Figure 6-4. Power Rail Sequence Timings (Sustain Well Power Management)

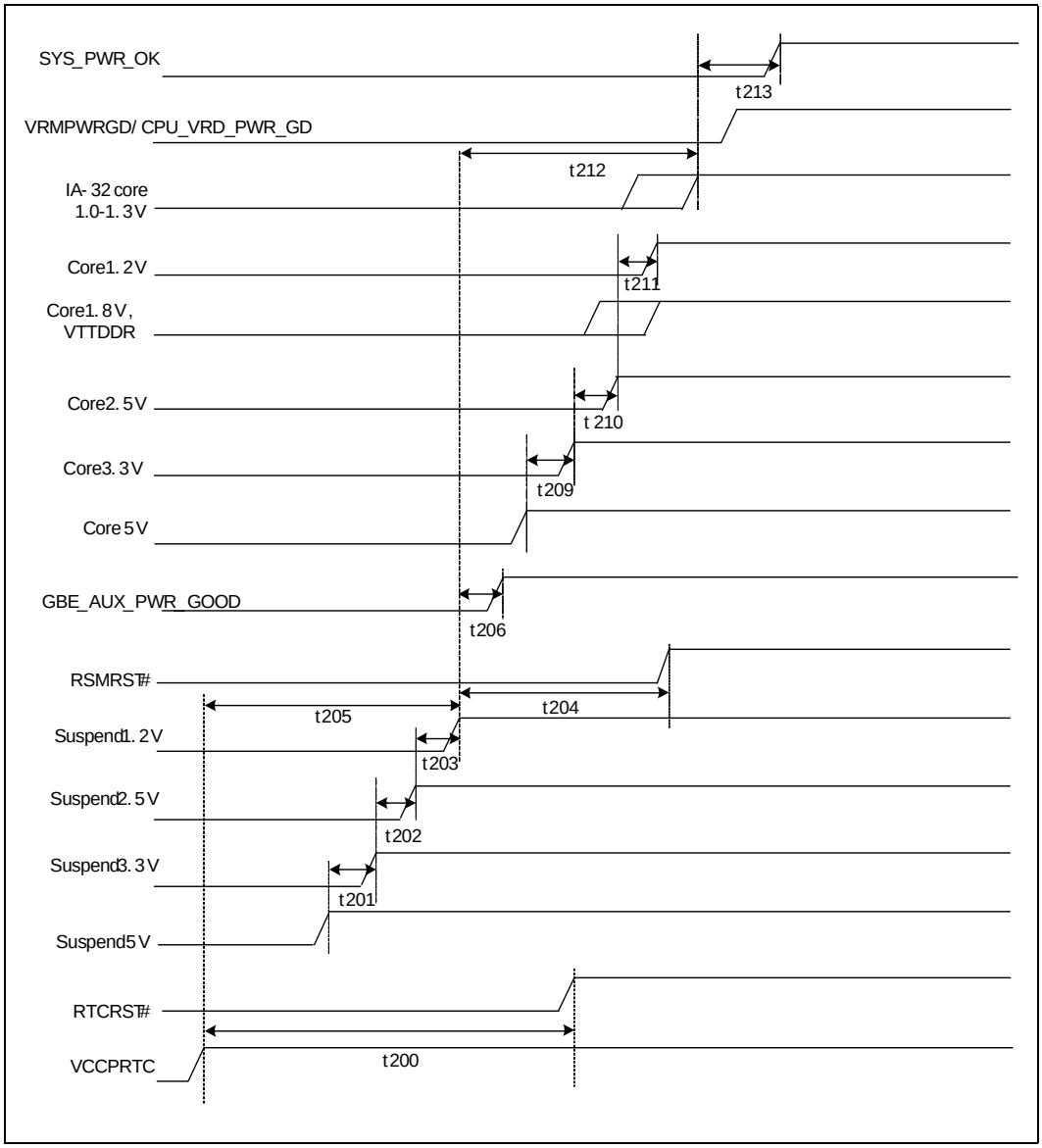


Figure 6-5. Power Rail Sequence Timing (No Sustain Well Power Management)

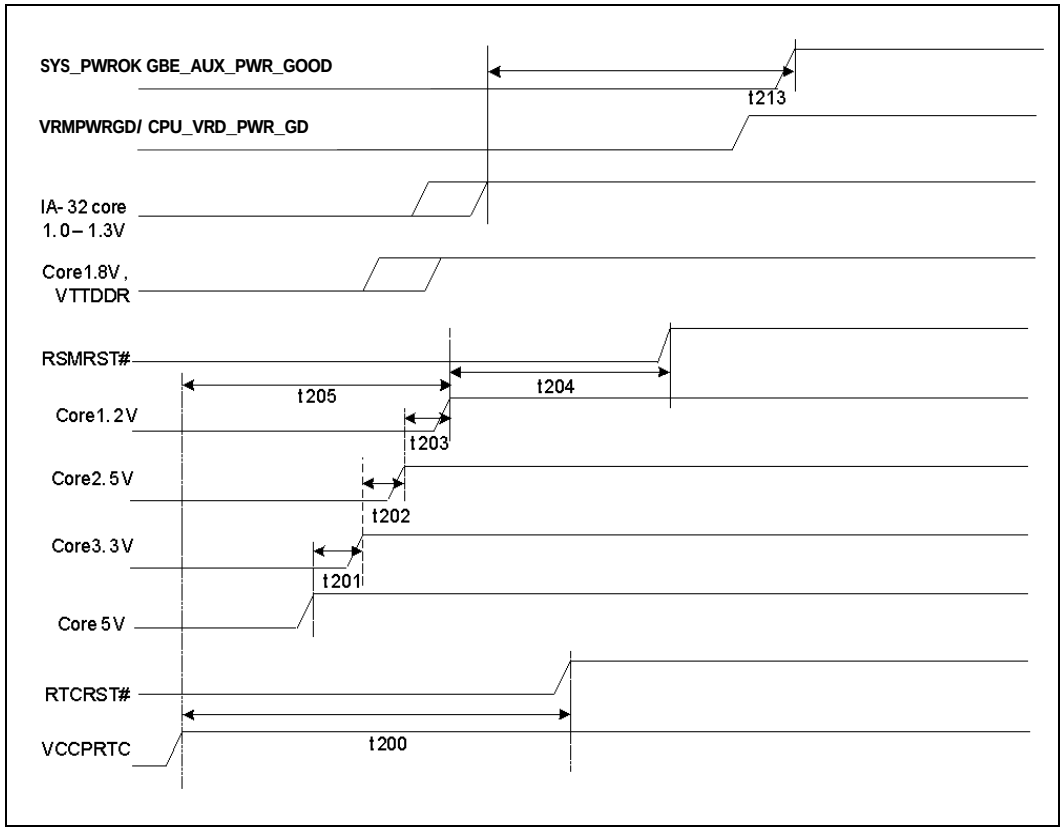


Table 6-4. Power Rail Sequence Signal Timings

Sym	Parameter	Min	Max	Units	Notes
t200	VCCPRTC active to RTCRST# inactive	18	–	ms	
t201	Suspend 5 V active to Suspend 3.3 V active	0	–	ms	1
t202	Suspend 3.3 V active to Suspend 2.5 V active	0	–	ms	2
t203	Suspend 2.5 V active to Suspend 1.2 V active	0	–	ms	3
t204	Suspend supplies active to RSMRST# inactive	10	–	ms	
t205	VCCPRTC supply active to Suspend supplies active	0	–	ms	4
t209	Core 5 V active to Core 3.3 V active	0	–	ms	1

NOTES:
1. The 5 V supply must power up before its associated 3.3 V supply within 0.3 V, and must power down after the 3.3 V supply within 0.3V.
2. Ensure the following: a) Suspend 3.3 V must power up before Suspend 2.5 V or after Suspend 2.5 within 0.3 V, b) Suspend 2.5 V must power down before Suspend 3.3 V or after Suspend 3.3 V within 0.3 V.
3. Ensure the following: a) Suspend 2.5 V must power up before Suspend 1.2 V or after Suspend 1.2 V within 0.3 V, b) Suspend 1.2 V must power down before Suspend 2.5 V or after Suspend 2.5 V within 0.3 V.
4. The VccSus supplies must never be active while the VCCPRTC supply is inactive.
5. Ensure the following a) Core 3.3 V must power up before Core 2.5 V or after Core 2.5 V within 0.3 V, b) Core 2.5 V must power down before Core 3.3 V or after Core 3.3 V within 0.3 V.
6. Ensure the following: a) Core 2.5 V must power up before Vcc1.2 V or after Core 1.2 V within 0.3 V, b) Core 1.2 V must power down before Core 2.5 V or after Core 2.5 V within 0.3 V.

Table 6-4. Power Rail Sequence Signal Timings

Sym	Parameter	Min	Max	Units	Notes
t210	Core 3.3 V active to Core 2.5 V active	0	–	ms	5
t211	Core 2.5 V active to Core 1.2 V active	0		ms	6
t212	Suspend supplies active to Core supplies active	0	–	ms	4
t213	All core supplies active to SYS_PWR_OK (platform signal) active	99	–	ms	

NOTES:
1. The 5 V supply must power up before its associated 3.3 V supply within 0.3 V, and must power down after the 3.3 V supply within 0.3V.
2. Ensure the following: a) Suspend 3.3 V must power up before Suspend 2.5 V or after Suspend 2.5 within 0.3 V, b) Suspend 2.5 V must power down before Suspend 3.3 V or after Suspend 3.3 V within 0.3 V.
3. Ensure the following: a) Suspend 2.5 V must power up before Suspend 1.2 V or after Suspend 1.2 V within 0.3 V, b) Suspend 1.2 V must power down before Suspend 2.5 V or after Suspend 2.5 V within 0.3 V.
4. The VccSus supplies must never be active while the VCCPRTC supply is inactive.
5. Ensure the following a) Core 3.3 V must power up before Core 2.5 V or after Core 2.5 V within 0.3 V, b) Core 2.5 V must power down before Core 3.3 V or after Core 3.3 V within 0.3 V.
6. Ensure the following: a) Core 2.5 V must power up before Vcc1.2 V or after Core 1.2 V within 0.3 V, b) Core 1.2 V must power down before Core 2.5 V or after Core 2.5 V within 0.3 V.

Figure 6-6. Powergood Reset Sequence

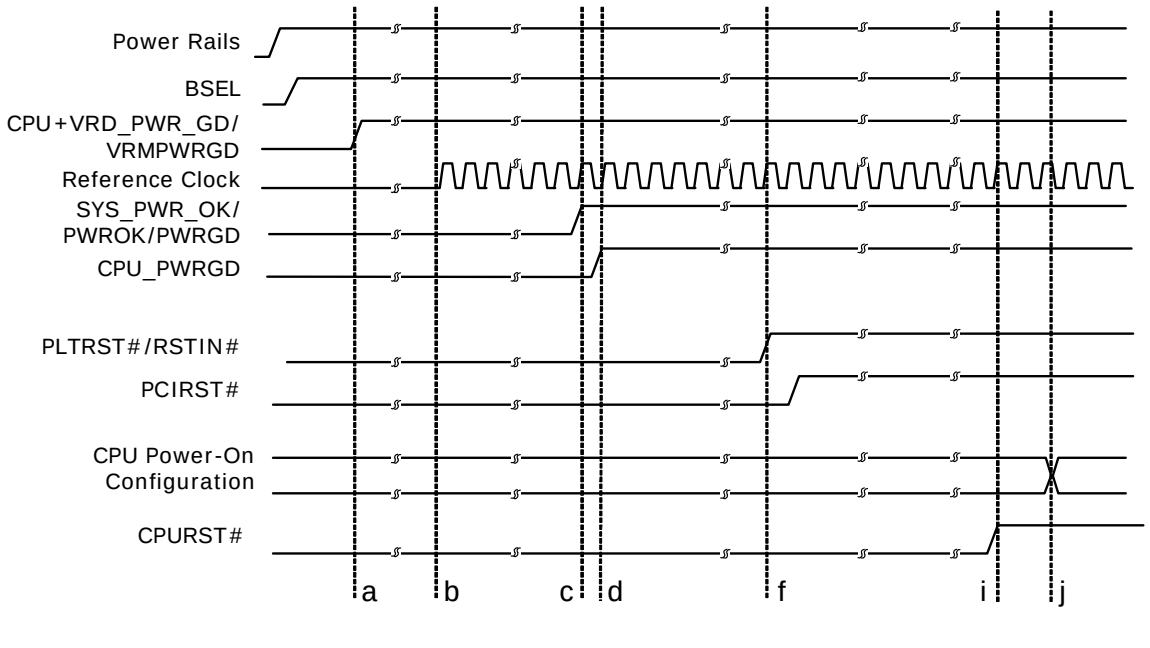


Table 6-5. Powergood Reset Timings

Timing	Description	Value
T_{ab}	VRMPWRGD/ CPU_VRD_PWR_GD assertion to Reference Clock Stable	2 ms
T_{ac}	VRMPWRGD/ CPU_VRD_PWR_GD assertion to SYS_PWR_OK (platform signal) assertion	99 ms
T_{cd}	SYS_PWR_OK (platform signal) assertion to CPU_PWRGD assertion (CPU_PWRGD is logical AND of VRMPWRGD/ CPU_VRD_PWR_GD and SYS_PWR_OK)	logic delay
Tdf	CPU_PWRGD assertion to PLTRST# de-assertion	1 ms
Tfi	RSTIN# deassertion to CPURST# (internal signal) de-assertion	1 ms + CPU_RST_DONE transaction delay (max 10,000 PCI-e clocks) + CPU_RST_DONE capture timer (min 2000 reference clocks)
Tij	CPURST# (internal signal) de-assertion to POC invalid	2 reference clocks

Figure 6-7. Hard Reset Sequence

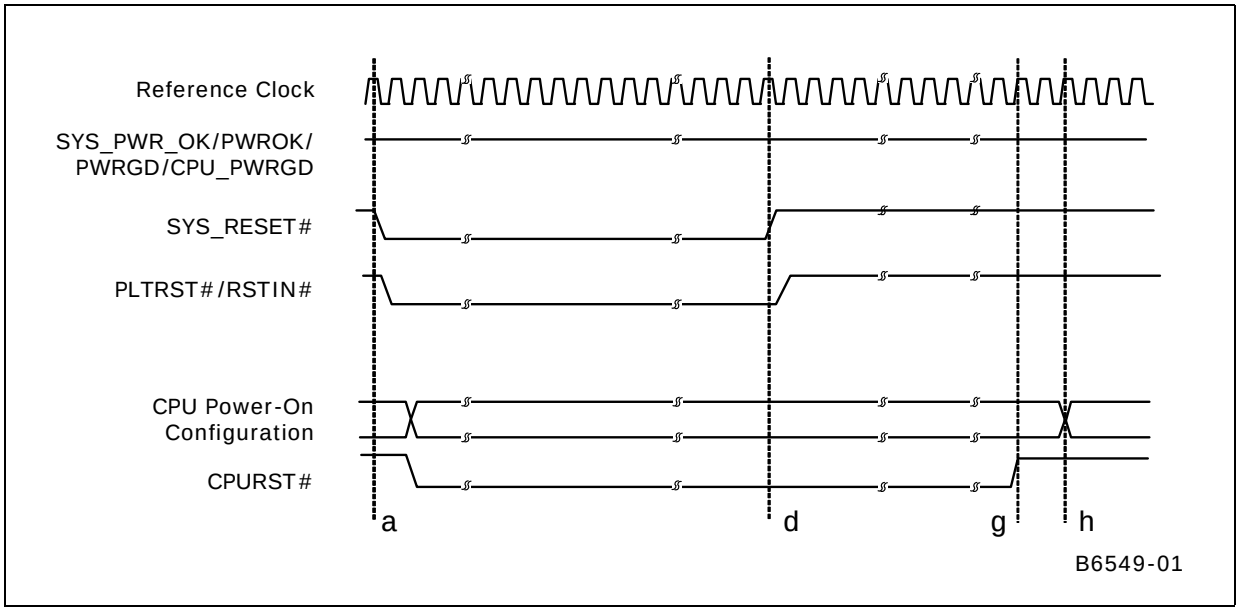


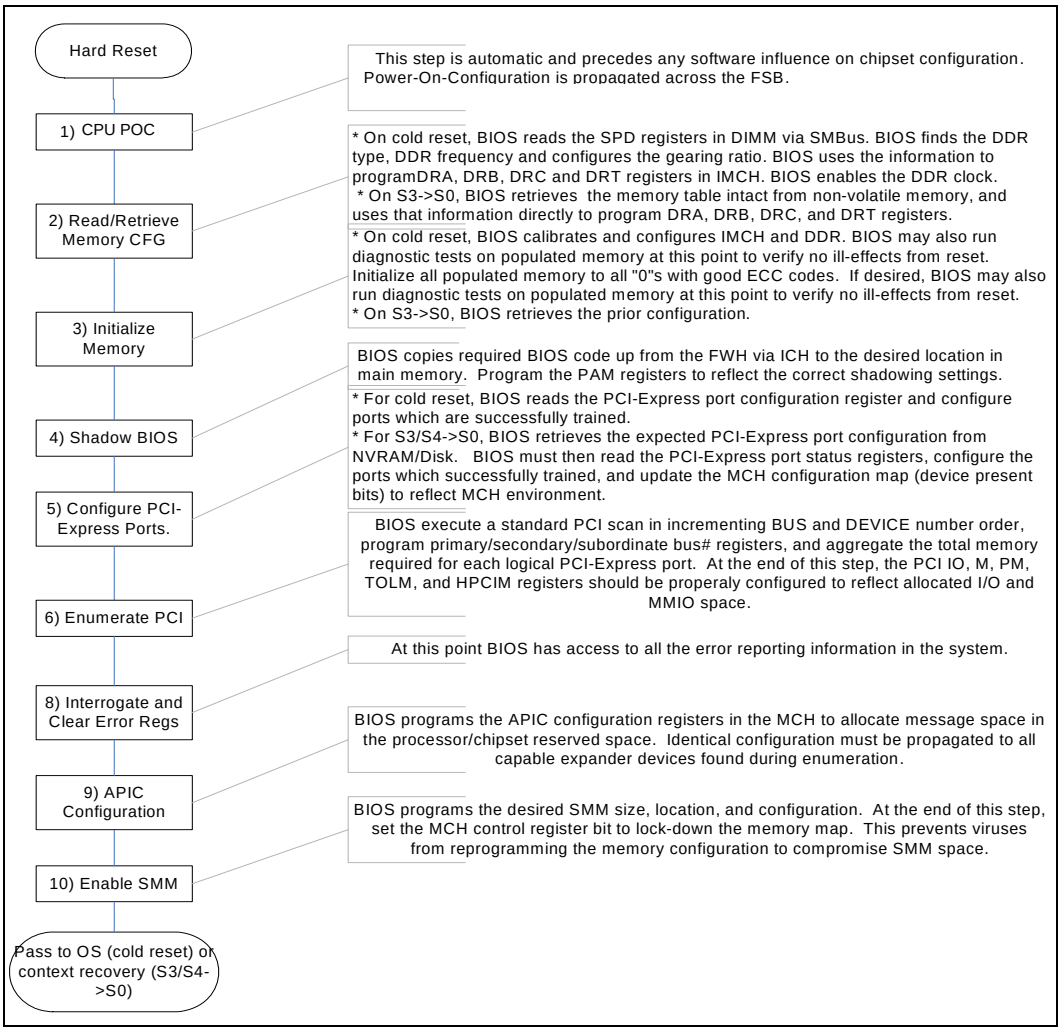
Table 6-6. Hard Reset Timings

Timing	Description	Value
T_{ad}	Minimum SYS_RESET assertion duration	100 ms
T_{dg}	RSTIN# deassertion to CPURST# (internal signal) de-assertion	1 ms + CPU_RST_DONE transaction delay (max 10,000 PCI-e clocks) + CPU_RST_DONE capture timer (min 2000 reference clocks)
T_{gh}	CPURST# (internal signal) de-assertion to POC invalid	2 reference clocks

6.2 BIOS Boot Flow (Initialization)

After hardware reset of the EP80579 and once IA -32 core has executed the reset micro-code, the IA 32 core resteers to reset vector (0xFFFF_FFF0) and starts fetching from BIOS code boot rom. BIOS starts execution from the reset vector regardless of whether wake is from S3, S4, or S5. To determine that it is S3 resume, BIOS checks the SUS_TYP field in the power management controller. If it is not S3, then normal boot occurs. If it is S3, then BIOS also checks the power failure bits (PWRBTNOR_STS, PWR_FLR, PWROK_FLR). If these are set to one, then memory contents cannot be relied on and normal boot is followed. If it is an S3 resume and the power failure bits are not set, then the S3 boot path is followed. The normal boot path is that used for cold reset and for S4/S5 resume. From a BIOS perspective there is no difference between S4 and S5. The following steps describe the boot sequence after reset before handover to OS.

Figure 6-8. BIOS Boot Flow (Cold Boot, S3/S4->S0)



6.2.1 Memory Configuration

As described in the flow chart, one of the first actions of the BIOS is to configure main memory. This section describes the memory configuration sequence of BIOS.

At reset the memory controller disables the output clock reference drivers for all DIMM slots, which prevents them from locking at the wrong frequency, and then relocking after this step in the initialization process.

Memory controller core logic boots at a factory set default frequency as is indicated by the BSEL Pin. The BIOS must first set the DDR frequency and ratio via the DRC register because the memory controller needs to re-lock itself at the desired DDR frequency.

BIOS configures the DDR frequency by reading the FSB frequency value from a processor internal MSR and DDR2 circuitry and sets the DDR Speed bits [3:0] of the DRC register (d0, f0, offset 7Ch).

Once the correct frequencies have been selected BIOS updates CKDIS (d0, f0, offset 8Ch), such that only populated ranks of populated DIMM slots receive output clocks from the memory controller.

Using the DDR SPD data, BIOS programs the IMCH DRA and DRB registers for proper translation of physical addresses to DDR row, bank, and column addresses.

BIOS further configures the DDR configuration, timing, and impedance compensation settings to enable reliable communication between the IMCH and the DDR DIMM devices. The DRT registers (d0, f0, offsets 78h and 64h) defaults settings can be found in Section 11.0, “System Memory Controller”.

BIOS further performs DDR calibration, memory initialization, and optional MemBIST.

At this stage BIOS is aware of the total amount of memory populated in the system, and may generate the starting value for the top of memory (TOM) register setting. BIOS will generally want to complete at least a rudimentary memory test sequence (next step) prior to finalizing the memory size information reported to the operating system.

For memory initialization details refer to Section 11.0, “System Memory Controller”.

Wake from S3/S4/S5 are also described in the Figure 6-8

6.2.2 Memory Initialization

At this point in the boot sequence memory contains random data from power-on, and would therefore generate non-deterministic ECC errors on read accesses. To zero-out memory and initialize all locations with good ECC, the memory controller provides a hardware engine which will walk all populated DRAM space issuing cache-line sized writes with all zeroes as data.

6.2.3 Boot from Network

Booting from network on Intel platforms is supported by PXE. PXE (Preboot eXecution Environment) is an existing open industry specification for network clients to automatically download software images and configuration parameters. The PXE client software is typically implemented as a BIOS Option ROM that is executed during the preboot phase of the client system. This Option ROM (OpROM) implements a sufficient network stack to perform all of the necessary network operations to boot an operating system. This OpROM image is written assuming IA-32 architecture and instruction set. Also, each OpROM image is modified specifically for a given Network Interface Controller (NIC).

The BIOS must first establish an environment suitable for execution of the PXE OpROM. This environment must provide the following:

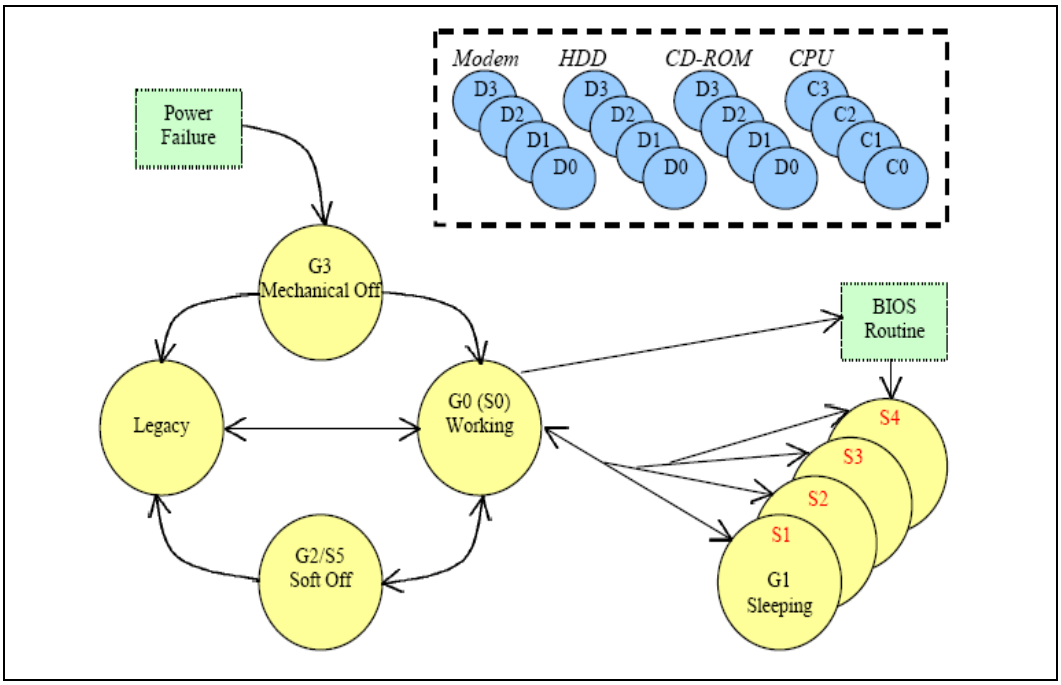
- System memory sufficient for C-code execution. This memory will be used to load/execute OpROM code, implement a call stack and data storage. In addition system memory must be available for loading/execution of OS.
- BIOS runtime services sufficient for OpROM execution. These services are typically provided by system BIOS and allow the PXE OpROM to be implemented for platform independence.
- The PCI sub-system is enumerated with allocation of system memory space, IO space and interrupts completed. This will establish the address location of resources accessed by the OpROM code during execution.

After OpROM initialization, the code remains in memory and waits until the BIOS reaches the point at which the OS-boot process begins. When a GbE is selected for boot, code in the OpROM will be used to initiate communication with a server to supply the images required for boot. After these images are loaded execution control passes from the BIOS to the images loaded in memory. This transition concludes the Preboot phase and enters in to the OS-boot phase.

6.3 Power Management

6.3.1 Power Management States

Figure 6-9. Global System Power States and Transitions



From a user-visible level, the system can be thought of as being in one of the states shown in Figure 6-9. In general use, computers alternate between the Working and Sleeping states. In the Working state, the computer is used to do work. User-mode application threads are dispatched and running. Individual devices can be in low-power (Dx) states and processors can be in low-power (Cx) states if they are not being used.

Any device the system turns off because it is not actively in use can be turned on with short latency. (What “short” means depends on the device. An LCD display needs to come on in sub-second times, while it is generally acceptable to wait a few seconds for a printer to wake.)

Precise definitions for Gx, Dx, Sx and Cx states are given in Table 6-7, Table 6-8, Table 6-9, and Table 6-10.

Table 6-7. Global Power States

Global Power State	Description
G3 Mechanical Off	A computer state that is entered and left by a mechanical means (for example, turning off the system’s power through the movement of a large red switch). Various government agencies and countries require this operating mode. It is implied by the entry of this off state through a mechanical means that no electrical current is running through the circuitry and that it can be worked on without damaging the hardware or endangering service personnel. The OS must be restarted to return to the Working state. No hardware context is retained. Except for the real-time clock, power consumption is zero.
G2/S5 Soft Off	A computer state where the computer consumes a minimal amount of power. No user mode or system mode code is run. This state requires a large latency in order to return to the Working state. The system's context will not be preserved by the hardware. The system must be restarted to return to the Working state. It is not safe to disassemble the machine in this state.
G1 Sleeping	A computer state where the computer consumes a small amount of power, user mode threads are not being executed, and the system “appears” to be off (from an end user's perspective, the display is off, and so on). Latency for returning to the Working state varies on the wake environment selected prior to entry of this state (for example, whether the system should answer phone calls). Work can be resumed without rebooting the OS because large elements of system context are saved by the hardware and the rest by system software. It is not safe to disassemble the machine in this state.
G0 Working	A computer state where the system dispatches user mode (application) threads and they execute. In this state, peripheral devices (peripherals) are having their power state changed dynamically. The user can select, through some UI, various performance/power characteristics of the system to have the software optimize for performance or battery life. The system responds to external events in real time. It is not safe to disassemble the machine in this state.

Table 6-8. Device States

Device State	Description
D3 Off	Power has been fully removed from the device. The device context is lost when this state is entered, so the OS software will reinitialize the device when powering it back on. Since device context and power are lost, devices in this state do not decode their address lines. Devices in this state have the longest restore times. All classes of devices define this state.
D2	The meaning of the D2 Device State is defined by each device class. Many device classes may not define D2. In general, D2 is expected to save more power and preserve less device context than D1 or D0. Buses in D2 may cause the device to lose some context (for example, by reducing power on the bus, thus forcing the device to turn off some of its functions).
D1	The meaning of the D1 Device State is defined by each device class. Many device classes may not define D1. In general, D1 is expected to save less power and preserve more device context than D2.
D0 Fully-On	This state is assumed to be the highest level of power consumption. The device is completely active and responsive, and is expected to remember all relevant context continuously.

Table 6-9. Sleeping States

Sleeping State	Description
S0	Fully Active
S1	The S1 sleeping state is a low wake latency sleeping state. In this state, no system context is lost (CPU or chip set) and hardware maintains all system context.
S2	The S2 sleeping state is a low wake latency sleeping state. This state is similar to the S1 sleeping state except that the CPU and system cache context is lost (the OS is responsible for maintaining the caches and CPU context). Control starts from the processor's reset vector after the wake event.
S3	The S3 sleeping state is a low wake latency sleeping state where all system context is lost except system memory. CPU, cache, and chip set context are lost in this state. Hardware maintains memory context and restores some CPU and L2 configuration context. Control starts from the processor's reset vector after the wake event.
S4	The S4 sleeping state is the lowest power, longest wake latency sleeping state supported by ACPI. In order to reduce power to a minimum, it is assumed that the hardware platform has powered off all devices. Platform context is maintained.
S5 Soft Off	The S5 state is similar to the S4 state except that the OS does not save any context. The system is in the “soft” off state and requires a complete boot when it wakes. Software uses a different state value to distinguish between the S5 state and the S4 state to allow for initial boot operations within the BIOS to distinguish whether or not the boot is going to wake from a saved memory image.

Table 6-10. CPU States

Processor Power State	Description
C0 - Full On	Processor core is active. All clocks are running. Processor can maintain cache coherency via snoops. Processor responds to interrupt.
C1 - Auto Halt	Processor Core is not active after executing an Auto-Halt instruction. Processor Core clock is internally gated. Processor can maintain cache coherency via snoops. Processor responds to interrupts. Aside from putting the processor in a non-executing power state, this state has no other software-visible effects.
C2 - Stop Grant	Processor Core is not active after its STPCLK# input is asserted. Processor Core clock is internally gated. Processor can maintain cache coherency via snoops. Processor responds to interrupts. Aside from putting the processor in a non-executing power state, this state has no other software-visible effects.
C3 - Deep Sleep	Processor Core is not active after its SLP# input is asserted. Processor Core clock is gated and PLLs are disabled. Processor does not respond to snoops or interrupts. While in the C3 state, the processor's caches maintain state but ignore any snoops. The operating software is responsible for ensuring that the caches maintain coherency. The EP80579 does not support C3 while functioning in S0 state, however, while in S1 state, The EP80579 will put the processor in Deep Sleep state through the assertion on the SLP# signal.

6.3.2 Power Management Support

As a system on a chip with embedded I/O devices and many SKUs, the EP80579 power management needs to perform multiple functions (typically under ACPI and/or BIOS system software control):

- Minimize power consumption of software-disabled interfaces/units.
- Transition between defined ACPI states as defined in Table 6-11.
- Support system wake-up from GPIO, PCI Express* devices and GbE MAC ports (Wake-on-LAN).

- Support cold and warm reset for the whole chip.

Table 6-11. ACPI States

ACPI State	IICH	IMCH	IA-32 core	AIOC
S0	Full On	Full On	C0, C1, C2	Full On
S1	Full On	Full On	Deep Sleep	Full On
S2	Not Supported	Not Supported	Not Supported	Not Supported
S3-hot	Not Supported	Not Supported	Not Supported	Not Supported
S3-cold (Suspend To RAM)	Sections stay on, Standby wells	DDR IO Power on but interface not active	Power Down	D3 (Stop Clock and Power Down, Except GbE MAC Wake-on LAN)
S4 (Suspend To Disk)	Sections stay on, Standby wells	Sections stay on, Standby wells	Power Down	Power Down
S5 Soft Off	Sections stay on, Standby wells	Sections stay on, Standby wells	Power Down	Power Down

Table 6-12. Power Wells Status for Supported ACPI States* (Sheet 1 of 2)

Power Well	Supply Pin(s)	S0	S1	S3-cold	S4	S5
Core	VTTDDR	On	On	On[a]	Off	Off
	VCCVC VCCA[1] VCCA[2]	On	On	Off	Off	Off
	VCC VCCUSB12 VCCAUSB12 VCCAHPLL VCCAPE0PLL12 VCCAPLL VCCAPE VCCARX VCCATX VCCRPE VCCSATA	On	On	Off	Off	Off
	VCC18 VCCTMP18 VCCAPE0PLL18	On	On	On[b]	Off	Off
	VCC25	On	On	Off	Off	Off
	VCC33 VCCGBE33 VCCSATA33 VCCABG3P3_USB VCCABGP033 VCCASATABG3P3	On	On	Off	Off	Off
	VCC50	On	On	Off	Off	Off

Table 6-12. Power Wells Status for Supported ACPI States* (Sheet 2 of 2)

Power Well	Supply Pin(s)	S0	S1	S3-cold	S4	S5
Suspend	VCC1P2_USBSUS VCCSUS1	On	On	On	On	On
	VCCSUS25	On	On	On	On	On
	VCCPSUS VCCGBEPSUS	On	On	On	On	On
	VCC50_SUS	On	On	On	On	On
RTC	VCCPRTC	On	On	On	On	On

a. VTTDR, can optionally be powered off during S3, but typically is derived from and tracks DDR IO voltage, VCC18, to avoid the complexity involved in timing the VTTDDR power up with the exit of S3.
b. VCCTMP18 and VCCAPE0PLL18 can optionally be powered off in S3 state. They are feed areas that consume very little power and are grouped with VCC18, which must be on in S3, to avoid requiring an additional power supply to support them.

6.3.2.1 Transitioning Between Power States

The EP80579 uses a cooperative power-down that is driven by software. To transition from S0 into the S3 or S4 state under ACPI/BIOS/OS/device driver control, system software is required to:

1. Suspend acceleration and security service application-level threads.
2. Place ASU and SSU devices into quiescent idle state, by completing all outstanding work requests, saving internal state to memory, then disabling ASU/SSU interrupts.
3. Quiesce I/O interfaces by disabling Rx of new traffic, finalizing outstanding Tx operations, disabling interrupts, and saving snap-shot of internal state to memory.
4. Save internal IMCH and IICH state to memory.
5. If transitioning to S4, move DRAM image to disk.
6. Drain all outstanding IMCH updates to DRAM.
7. Signal IICH to power-down the IA-32 core, the AIOC, the IMCH and the IICH. In case of S3, the memory interface is placed into self-refresh mode. In S3/S4 state, the GbE MAC, GPIO and LPC interfaces remain powered to process wake events.

An external wake event from GPIO, GbE MAC (received Wake-on-LAN packet) or PCI Express* signals the IICH to initiate a complete reset sequence that transitions the EP80579 back into the S0 state as follows:

1. All internal states outside of the IICH Resume Well are fully reset.
2. On resume from S3, BIOS does not reinitialize memory. Please refer to Section 6.2, “BIOS Boot Flow (Initialization)” for details.
3. On resume from S3 or S4, all EP80579 device drivers (including ASU and SSU drivers) are expected to restore internal device state from their memory resident save area.
4. On resume from S3 or S4, resume acceleration and security service application-level threads.

Note: There is no support for wake from USB when in S3/S4/S5.

6.3.2.2 Power State Transition Timing Diagrams

For power state transition timing details, refer to Section 49.5.1.2, “Power Management AC Characteristics”.

6.3.3 Thermal Sensor

The EP80579 has an on-die thermal sensor. It helps to control the silicon temperature by monitoring the silicon thermal status and activating the thermal control sequence when the silicon reaches its maximum operating temperature. See Chapter 50.0, “Thermal Specifications and Design Considerations” for details.

6.3.4 ACPI Implementation

The PCI device tree for the EP80579 is shown in Figure 3-4, “Overview of PCI infrastructure for On-die Devices” on page 121. The EP80579 follows the ACPI specification (http://www.acpi.info/DOWNLOADS/ACPIspec30a.pdf) which implies that all the PCI devices implement the standard PCI/ACPI registers:

1. PCI Power Management Block
2. Capability ID
3. Next Item Pointer
4. Power Management Capabilities (PMC)
5. Power Management Control and Status Register (PMCSR)

§ §

7.0 Register Summary

7.1 Overview of Register Descriptions and Summaries

This chapter presents summary tables for the registers and MMIO spaces that the components of the EP80579 define. In addition, this chapter describes how to read the standard register description that is used throughout this document to describe the functionality of individual registers.

The register summaries in this document follow a general formatting structure that includes, for each register, its name, default value, offsets, and a cross-reference to its detailed register description. Locations that are not associated with a register in the summary table should be assumed to be reserved.

The summaries in this chapter are organized by functional unit and describe the EP80579 registers that are visible from the IA platform perspective (e.g., PCI registers, PCI memory-mapped I/O registers, fixed IA I/O space registers, etc.).

7.1.1 Register Description Tables

In addition to the summaries in this chapter, this document uses a standard tabular format to describe the operation of each register in the device. These descriptions are cross-referenced from summary tables and cover the specific content and functionality of a register. The information in a register description table can be broken down into three major areas:

- Materialization information that establishes how the register appears to software.
- Global information that lists the size, default, value, power well, etc. for the register.
- Field definitions that list the name, description, default value, and attributes of all the fields in the register.

The register definition can describe a unique register entity in the design or serve as a template that describes several register entities are instantiated in the design. The materialization information in the register description table can handle common scenarios with minimal duplication of content:

- A single physical register that materializes at multiple “addresses” in the system (i.e., a double- or triple-mapped register).
- Multiple physical registers that share the same definition but materialize within different “device” instances.
- Multiple physical registers that share the same definition but materialize repeatedly within a single “device”.

The register description table format handles the first two scenarios through “views” that make up the bulk of the materialization information in a register description table and handles the final scenario through a set of “repeated register” conventions.

The materialization information in a table includes specification of one or more “views”. Each view consists of a view “type” along with several type-specific fields that serve to specify the “address” of the register in the system from a particular perspective. For example, a PCI “view” includes, in part, the PCI bus number as a parameter since this information is necessary to specify the location of the register. In a register description table, each view specification occupies one row of the table. A register description table includes one or more views depending on how the register materializes to software.

There are a number of different views that this document uses to describe how EP80579 registers materialize to software. Table 7-1 defines the views that this document uses along with the type-specific fields that each view includes.

Table 7-1. Definition of the Views Used in Register Description Tables

View Type	Describes Registers in	Type-Specific Fields	
		Name	Description
PCI	PCI configuration space or memory/ IO spaces that are mapped via PCI BARs	B:D:F	PCI Bus, Device, and Function number that the register is associated with through PCI configuration, memory-mapped, or I/O-mapped spaces (see BAR).
		BAR	PCI base address register in B:D:F that the register is referenced from. This field is “Configuration” for registers that materialize in PCI configuration space. Otherwise, the field is the name of the BAR register in B:D:F that provides the base address. The register materializes in memory space unless the BAR field contains “(IO)”; i.e., views with a BAR of “FOOBAR” and “FOOBAR (IO)” materialize in memory and I/O spaces, respectively.
		Offset Start	Starting offset from BAR. The offset is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].
		Offset End	Ending offset of register from BAR. The offset is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].
IA F	General “fixed” location in IA memory or I/O spaces	Base Address	Base address. Typically, this field contains a number or register name. It may contain a comma-separated list if the register can materialize at one of several possible bases (for example, “100h, 200h based on FOOREG”). The register materializes in memory space unless the base address field contains “(IO)”; i.e., views with a base address of “0000h” and “0000h (IO)” are in memory and I/O space, respectively.
		Offset Start	Starting address or offset from base address field. The offset is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].
		Offset End	Ending address or offset from base address field. The offset is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].

Table 7-1. Definition of the Views Used in Register Description Tables

View Type	Describes Registers in	Type-Specific Fields	
		Name	Description
IA I	Indirect location accessed via index/window register pair	Win:Idx	Name of the window and index registers that expose the indirect register(s). For example, W:I of “FOO:BAR” indicates that accesses to FOO indirectly access the register that is located at the index (i.e., offset) in BAR.
		Offset Start	Starting offset to put in the index register to access the indirect register. The index is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].
		Offset End	Ending address or offset from base address field. The offset is in bytes unless the field contains “(2B)”, “(4B)”, and “(8B)” to indicate a single-, double-, or quad-word offset, respectively[a].

a. In this usage, words, double words, and quad words are 16-, 32-, and 64-bits, respectively.

The register description tables adopt the convention that Table 7-2 and Table 7-3 describe for view and register name information to distinguish the three scenarios mentioned above (single physical register, multiple physical registers in the same device, and multiple physical registers in different devices). Table 7-2 describes how to interpret the combination of register views and name. This table elides the type-specific fields and simply lists the types of views for brevity.

Table 7-2. View Convention to Describe Single Versus Multiple Physical Registers

Scenario	Register Name	Example Views from Register Table	Interpretation
Single	FOO	PCI PCI IA F IA F	This example shows a single physical register FOO. FOO materializes in two different PCI devices and at two different “fixed” memory locations that the PCI and IA M views describe.

Table 7-2. View Convention to Describe Single Versus Multiple Physical Registers

Scenario	Register Name	Example Views from Register Table	Interpretation
Multiple (Different Devices, Same name)	BAZ	PCI 1 PCI 2 IA F 1 IA F 2	All views for the register include a numeric suffix that corresponds to the physical instance of the register. Views with the same suffix address the same physical register. This example shows two physical registers matching the description of BAZ. The first instance of BAZ materializes in the PCI device and at the memory location that the PCI 1 and IA F 1 views describe, while the second instance of BAZ materializes in PCI 2 and at IA F 2.
Multiple (Different Devices and Names)	BAZ{2:0}	PCI 0 PCI 1 PCI 2 IA F 0 IA F 1 IA F 2	Like the Multiple scenario above, the views also include a numeric suffix that, in this case, maps to different register names. The {m:n} notation in the register name indicates that each physical instance of the register has a different name that includes the instance number. An instance number i is an integer such that $n \leq i \leq m$. This implies that there are m - n + 1 distinct instances of the register. This example shows three physical registers matching the description of BAZ: BAZ0, BAZ1, and BAZ2. BAZ0 materializes in the PCI device and at the memory location that the PCI 0 and IA F 0 views describe, BAZ1 materializes in PCI 1 and at IA F 1, and BAZ2 materializes in PCI 2 and at IA F 2.
Multiple (Same Device)	BLEH[1-3]	PCI PCI IA F IA F	The register name has a suffix indicating the number of physical instances of the register. The format of the suffix is “[m-n]” where m and n are integers and implies that there are n - m + 1 distinct instances of the register. This example shows three physical registers matching the description of BLEH: BLEH[1], BLEH[2], and BLEH[3]. Each register materializes in two different PCI devices and at two different “fixed” memory locations that the PCI and IA M views describe.

When describing multiple physical registers of the same “format” that materialize in the same device, the tables use the convention that Table 7-3 describes to denote the offsets[1].

Table 7-3. Offset Convention to Describe Multiple Physical Registers in the Same Device

Register Name	Offset in View(s) (Start or End)	Interpretation
BLEH[1-3]	10h, 38h, 70h	Three BLEH registers whose offsets match the elements of the comma-separated list. The offsets are BLEH[1] = 10h, BLEH[2] = 38h, BLEH[3] = 70h.
BAZ[1-2]	103h at 2h	Two BAZ registers whose offsets stride by 2h starting from 103h. The offsets are BAZ[1] = 103h, BAZ[2] = 105h.

The remainder of this section presents several examples that illustrates how to read register definition tables. These examples are intended to illustrate how to interpret a register definition table, *not* describe actual registers in the EP80579. As a result, some of the examples may be contrived from the perspective of an implementation.

1. In this scenario, it is the offset that distinguishes the different materialization points of the registers; the remaining view fields should be the same since this scenario applies to registers in the same “device”.

Given the conventions outlined above, Table 7-4 presents an example definition for the register EG_SINGLE which corresponds to a single physical register. This register materializes at offset E0h in the configuration space for PCI device 4, at *double-word* (i.e., 32-bit) offset `100h` in the memory region defined by FOOBAR of PCI device 12, and at the fixed offset `0A0h` in IA I/O space[1].

Table 7-4. EG_SINGLE: Example Single Register with Different Views

Description: A single physical register that materializes at multiple locations.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** E0h **Offset End:** E1h	
View: PCI	**BAR:** FOOBAR		**Bus:Device:Function:** 0:12:0	**Offset Start:** 100h (4B) **Offset End:** 101h (4B)	
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** A0h **Offset End:** A1h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	MAGIC	**Magic Stuff:** This field contains a magic number, 8086h.		8086h	RO

In the table, the materialization information includes the rows starting with “View” and the title; the field definitions includes all rows below the row starting with “Bit Range”, and the rows beginning “Description” and “Size” encompass the global information.

Table 7-5 presents an example definition for the register EG_MULTI_DIFF which corresponds to two physical registers that materialize in different “devices”. The first instance materializes at offset `D0h` in the configuration space for PCI device 20 and at the fixed offset `FFEF01B0h` in IA memory space. The second instance materializes at offset `D0h` in the configuration space for PCI device 21 and at the fixed offset `FFEF11C0h` in IA memory space. Both registers have the same name, EG_MULTI_DIFF.

Table 7-5. EG_MULTI_DIFF: Example Multiple Registers in Different Devices with Different Views

Description: A set of two physical registers that materialize in different devices.					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:20:0	**Offset Start:** D0h **Offset End:** D3h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:21:0	**Offset Start:** D0h **Offset End:** D3h	
View: IA F 1	**Base Address:** 00000000h			**Offset Start:** FFEF01B0h **Offset End:** FFEF01B3h	
View: IA F 2	**Base Address:** 00000000h			**Offset Start:** FFEF11C0h **Offset End:** FFEF11C3h	
Size: 32 bit	**Default:** DEADBEEFh			**Power Well:** Core, Reset	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	BMAGIC	**Black Magic Stuff:** This field contains a magic number.		DEADBEEFh	RO

1. Recall that offsets are in bytes unless otherwise specified with “(2B)”, “(4B)”, or “(8B)”.

This table illustrates how the global information can be tied to a particular register in multiple register cases. In this table, the first and second instances of EG_MULT_DIFF are in the “Core” and “Reset” power wells, respectively. Typically, this notation is only be used for the power well.

It is also possible to incorporate the instance number in the name as Table 7-2 describes. Using the name EG_MULTI{2:1}_DIFF in Table 7-5 would identify two physical registers with different names. The first, EG_MULTI2_DIFF, materializes according to the PCI 2 and IA F 2 views, while the second, EG_MULTI1_DIFF, materializes according to the PCI 1 and IA F 1 views.

Table 7-6 presents an example definition for the register EG_MULTI_SAME which corresponds to two physical registers that materialize in the same “device”. The instance EG_MULT_SAME[1] materializes at offset ACE0h in the I/O region defined by BLAHBAR of PCI device 16 and at the fixed offset 100h in IA I/O space. The instance EG_MULT_SAME[2] materializes at offset ACE4h in the I/O region defined by BLAHBAR of PCI device 16 and at the fixed offset 120h in IA I/O space.

Table 7-6. EG_MULTI_SAME[1-2]: Example Multiple Registers in Same Device with Different Views

Description: A set of two physical registers that materialize in the same device.					
View: PCI	**BAR:** BLAHBAR (IO)	**Bus:Device:Function:** 0:16:0		**Offset Start:** ACE0h at 4h **Offset End:** ACE3h at 4h	
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 100h, 120h **Offset End:** 103h, 123h	
Size: 32 bit	**Default:** DEAD8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	BRMAGIC	**Blacker Magic Stuff:** This field contains a magic number.		DEAD8086h	RO

Note that in this example, the strides do not need to be the same across the different views.

Finally, Table 7-7 presents an example definition for the register EG_INDEX which corresponds to a physical registers that materialize indirectly. To access the EG_INDEX register, the starting offset 0100h (in double words of 32-bits) is written to the APIC_IDX index register to select EG_INDEX which is then accessed through the APIC_WND window registers.

Table 7-7. EG_INDEX: Example Single Indexed Register

Description: A single indexed register.					
View: IA I	**Win:Idx** APIC_WND:APIC_IDX			**Offset Start:** 0100h (4B) **Offset End:** 0100h (4B)	
Size: 32 bit	**Default:** 0ACEFACEh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	MAGIC	**Magic Stuff:** This field contains a magic number.		0ACEFACEh	RO

The offsets should always fit within the index register.

7.1.2 Register Field Access Attributes

Table 7-8 describes the register field access attribute acronyms that this document uses in its summaries and descriptions.

Table 7-8. Register Field Access Attributes

Attribute	Description
RV	Reserved – A reserved field.
RO	Read-Only – Software/BIOS can only read this bit. Contents are either hardwired or set by hardware.
WO	Write-Only – Not supported as a bit. The write causes a hardware event to take place.
WRC	Write/Read to Clear – Writes and reads clear. See bit descriptions.
RC	Read to Clear – Cleared automatically when read.
RW	Read/Write – Software/BIOS can read and write this bit.
RWC	Read/Write-Clear – Software/BIOS can read this bit and must write to a 1 to clear this bit.
RCWC	Read-Clear/Write-Clear – Cleared when read or Write one to clear.
RW0C	Read/Write zero to Clear – Software/BIOS can read this bit, and must write to a 0 to clear this bit.
R0/W	Read zero/Write – This register will only read 0. Must read register description for write actions.
RO/RWC	Read only/Read-Write-Clear – Attribute dependent on configuration. See bit description
RS/W1C	Read-Set/Write-Clear – Read to set, write 1 to clear
RWS	Read/Write-Set – Software/BIOS can read this bit and write it to a 1. Hardware clears this bit.
RWL	Read/Write-Lock – Software/BIOS can read and write this bit. Hardware or another configuration bit can lock this bit and prevent it from being updated.
RWO	Read/Write-Once – Software/BIOS can read this bit, but can only write this bit once. It is a special form of RWL. Once any byte within a register with RWO bits has been written, the RWO bits are locked and only a reset can clear its contents. Any exceptions are clearly documented.

In some cases, the access attribute for a field may depend on a setting of a fuse or a value in another register. In this case, the access types for the field will be of the form “X or Y”. Readers should consult the description of the field for additional information on the conditions that enable the various access types. For example, a field FOO that is either RO or RW based on the setting of a fuse bit would have an access type of “RO or RW”.

7.1.3 Register Nomenclature and Values

The summary and description tables also include register and field values. In general, a number is given by a string of digits followed by a single-character that identifies the base as decimal, “d”, hexadecimal, “h”, or binary, “b”. Both the digits and base pieces of a number are always case-insensitive[1]. That is, beefh and BEEFH represent the same hexadecimal number. A value of 0 (zero) does not require a base character. If a value is non-zero, the default base is assumed to be binary unless stated otherwise. All hexadecimal and decimal numbers must always have a base character.

In addition to the legal digits for the base, the digit string can contain “x” to denote undefined values. Typically, Xh is equivalent to XXXXb, XXh is equivalent to XXXXXXXXb. However, a 5-bit wide field that is undefined can be represented as both XXh and XXXXXb; that is, a nibble in a hex representation must be “x” if any of its bits are “x”.

1. The “suggested” convention is that digits are always upper case while the base is always lower case.

There are three special values that can be used in place of numbers to indicate cases where the value of a register or field is not fixed at design time and depends on other parameters:

- **Fuse:** The setting of one or more fuses determines the value of the register or field.
- **Strap:** The setting of a strap pin determines the value of the register or field.
- **Variable:** The setting of other on-die state determines the value of the register or field.

In these cases, the description of the register or field that is not fixed should contain additional information that describes how the EP80579 arrives at the value.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

7.1.4 "Sticky" Register Fields

For each field in a register, the description tables include an indication of whether or not the bits that make up the field are "sticky" or not. Sticky bits in registers will retain their value across a hard reset. Resetting field in a register that is designated as "sticky" to its default/reset value requires a cold reset. Unless explicitly noted, all register fields are assumed to not be sticky.

7.2 IA-32 core Registers

The IA-32 core registers are described in the related documentation for the Intel® Pentium® M Processor .

7.3 IMCH and IICH Registers

This section summarizes the registers found in the IMCH and the IICH.

7.3.1 IMCH Registers: Bus 0, Device 0, Function 0

The IMCH includes the registers listed in Table 7-9 through Table 7-11. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 16.1, "IMCH Registers: Bus 0, Device 0, Function 0", and Section 16.7, "Memory Mapped I/O for NSI Registers" for detailed discussion of these registers.

Table 7-9. Bus 0, Device 0, Function 0: Summary of IMCH PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID – Vendor Identification Register" on page 391	8086h
02h	03h	"Offset 02h: DID – Device Identification Register" on page 391	5020h
04h	05h	"Offset 04h: PCICMD: PCI Command Register" on page 392	0006h
06h	07h	"Offset 06h: PCISTS: PCI Status Register" on page 393	0010h
08h	08h	"Offset 8h: RID - Revision Identification Register" on page 394	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 394	00h
0Bh	0Bh	"Offset 0Bh: BCC – Base Class Code Register" on page 394	06h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 395	80h
14h	17h	"Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 396	00000000h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 396	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 397	0000h
4Ch	4Fh	"Offset 4Ch: NSIBAR - Root Complex Block Address Register" on page 397	00000000h
50h	50h	"Offset 50h: CFG0- IMCH Configuration 0 Register" on page 398	0Ch
51h	51h	"Offset 51h: IMCH_CFG1 – IMCH Configuration 1 Register" on page 399	00000h
53h	53h	"Offset 53h: CFGNS1 - Configuration 1 (Non-Sticky) Register" on page 399	00h
58h	58h	"Offset 58h: FDHC - Fixed DRAM Hole Control Register" on page 400	00h
59h	59h	"Offset 59h: PAM0 - Programmable Attribute Map 0 Register" on page 401	00h
5Ah	5Ah	"Offset 5Ah: PAM1: Programmable Attribute Map 1 Register" on page 402	00h
5Bh	5Bh	"Offset 5Bh: PAM2 - Programmable Attribute Map 2 Register" on page 403	00h
5Ch	5Ch	"Offset 5Ch: PAM3 - Programmable Attribute Map 3 Register" on page 404	00h
5Dh	5Dh	"Offset 5Dh: PAM4 - Programmable Attribute Map 4 Register" on page 405	00h
5Eh	5Eh	"Offset 5Eh: PAM5 - Programmable Attribute Map 5 Register" on page 406	00h
5Fh	5Fh	"Offset 5FH: PAM6 - Programmable Attribute Map 6 Register" on page 407	00h
9Ch	9Ch	"Offset 9Ch: DEVPRES - Device Present Register" on page 408	33h
9Dh	9Dh	"Offset 9Dh: EXSMRC - Extended System Management RAM Control Register" on page 409	00h
9Eh	9Eh	"Offset 9Eh: SMRAM - System Management RAM Control Register" on page 411	02h
9Fh	9Fh	"Offset 9Fh: EXSMRAMC - Expansion System Management RAM Control Register" on page 413	07h

Table 7-9. Bus 0, Device 0, Function 0: Summary of IMCH PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
B8h	BBh	"Offset B8h: IMCH_MENCBASE: IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Base Address Register" on page 413	000FFFFFh
BCh	BFh	"Offset BCh: IMCH_MENCLIMIT - IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Limit Address Register" on page 414	00000000h
C4h	C5h	"Offset C4h: TOLM - Top of Low Memory Register" on page 415	0800h
C6h	C7h	"Offset C6h: REMAPBASE - Remap Base Address Register" on page 416	03FFh
C8h	C9h	"Offset C8h: REMAPLIMIT – Remap Limit Address Register" on page 416	0000h
CAh	CBh	"Offset CAh: REMAPOFFSET - Remap Offset Register" on page 417	0000h
CCh	CDh	"Offset CCh: TOM - Top Of Memory Register" on page 417	0000h
CEh	CFh	"Offset CEh: HECBASE - PCI Express Port A (PEA) Enhanced Configuration Base Address Register" on page 418	E000h
D8h	D8h	"Offset D8h: CACHECTL0 - Write Cache Control 0 Register" on page 418	00h
DEh	DFh	"Offset DEh: SKPD - Scratchpad Data Register" on page 419	0000h
F6h	F6h	"Offset F6h: IMCH_TST2 - IMCH Test Byte 2 Register" on page 419	00h
60h at 1h	60h at 1h	"Offset 60h: DRB[0-3] - DRAM Row [3:0] Boundary Register" on page 421	ffh
70h at 4h	73h at 4h	"Offset 70h: DRA[0-1] - DRAM Row [0:1] Attribute Register" on page 422	00000515h
78h	7Bh	"Offset 78h: DRT0 - DRAM Timing Register 0" on page 424	242AD280h
64h	67h	"Offset 64h: DRT1 - DRAM timing Register 1" on page 431	12110000h
7Ch	7Fh	"Offset 7Ch: DRC - DRAM Controller Mode Register" on page 435	00000002h
84h	87h	"Offset 84h: ECCDIAG - ECC Detection/Correction Diagnostic Register" on page 437	00000000h
88h	8Bh	"Offset 88h: SDRC - DDR SDRAM Secondary Control Register" on page 439	00000002h
8Ch	8Ch	"Offset 8Ch: CKDIS - CK/CK# Clock Disable Register" on page 441	00h
8Dh	8Dh	"Offset 8Dh: CKEDIS - CKE Clock Enable Register" on page 442	00h
90h	93h	"Offset 90h: SPARECTL - SPARE Control Register" on page 443	00000000h
B0h	B3h	"Offset B0h: DDR2ODTC - DDR2 ODT Control Register" on page 444	00000000h

Table 7-10. Bus 0, Device 0, Function 0: Summary of IMCH Configuration Registers Mapped Through NSIBAR Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: SNSIVCECH - NSI Virtual Channel Enhanced Capability Header Register" on page 680	04010002h
04h	07h	"Offset 04h: NSIPVCCAP1 - NSI Port VC Capability Register 1" on page 680	00000000h
08h	0Bh	"Offset 08h: NSIPVCCAP2 - Port VC Capability Register 2" on page 681	00000001h
0Ch	0Dh	"Offset 0Ch: NSIPVCCTL - NSI Port VC Control Register" on page 682	0000h
10h	13h	"Offset 10h: NSIVC0RCAP - NSI VC0 Resource Capability Register" on page 682	00000001h
14h	17h	"Offset 14h: NSIVC0RCTL - NSI VC0 Resource Control Register" on page 683	800000FFh
1Ah	1Bh	"Offset 1Ah: NSIVC0RSTS - NSI VC0 Resource Status Register" on page 684	0002h

Table 7-10. Bus 0, Device 0, Function 0: Summary of IMCH Configuration Registers Mapped Through NSIBAR Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
80h	83h	"Offset 80h: NSIRCILCECH - NSI Root Complex Internal Link Control Enhanced Capability Header Register" on page 684	00010006h
84h	87h	"Offset 84h: NSILCAP - NSI Link Capabilities Register" on page 685	0003A041h

Table 7-11. Bus 0, Device 0, Function 0: Summary of IMCH SMRBASE Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: NOTESPAD - Note (Sticky) Pad for BIOS Support Register" on page 601	0000h
02h	03h	"Offset 02h: NOTEPAD - Note Pad for BIOS Support Register" on page 601	0000h
40h	43h	"Offset 40h: DCALCSR – DCAL Control and Status Register" on page 602	00000000h
44h	47h	"Offset 44h: DCALADDR - DCAL Address Register" on page 606	00000000h
48h at 1h	48h at 1h	"Offset 48h: DCALDATA[0-71] - DRAM Calibration Data Register" on page 607	00000000h
94h	96h	"Offset 94h: RCVENAC - Receiver Enable Algorithm Control Register" on page 611	180810h
98h	9Bh	"Offset 98h: DSRETC - DRAM Self-Refresh (SR) Extended Timing and Control Register" on page 611	5c141400h
9Ch	9Ch	"Offset 9Ch: DQSFAIL1 - DQS Failure Configuration Register 1" on page 612	00h
A0h	A3h	"Offset A0h: DQSFAIL0 - DQS Failure Configuration Register 0" on page 613	00000000h
A4h	A7h	"Offset A4h: DRRTC00 - Receive Enable Reference Output Timing Control Register" on page 615	06060606h
A8h	ABh	"Offset A8h: DRRTC01 - Receive Enable Reference Output Timing Control Register" on page 616	06060606h
C4h	C4h	"Offset C4h: DRRTC02 - Receive Enable Reference Output Timing Control Register" on page 616	06h
B4h	B7h	"Offset B4h: DQSOFCS00 - DQS Calibration Register" on page 617	00000000h
B8h	BBh	"Offset B8h: DQSOFCS01 - DQS Calibration Register" on page 617	00000000h
C6h	C6h	"Offset C6h: DQSOFCS02 - DQS Calibration Register" on page 618	00h
BCh	BFh	"Offset BCh: DQSOFCS10 - DQS Calibration Register" on page 618	00000000h
C0h	C3h	"Offset C0h: DQSOFCS11 - DQS Calibration Register" on page 619	00000000h
C7h	C7h	"Offset C7h: DQSOFCS12 - DQS Calibration Register" on page 619	00h
CCh	CFh	"Offset CCh: WPTRTC0 - Write Pointer Timing Control Register" on page 620	00000000h
D0h	D0h	"Offset D0h: WPTRTC1 - Write Pointer Timing Control 1 Register" on page 621	00h
D4h	D7h	"Offset D4h: DDQSCVDP0 - DQS Delay Calibration Victim Pattern 0 Register" on page 621	aaaa0a05h
D8h	DBh	"Offset D8h: DDQSCVDP1 - DQS Delay Calibration Victim Pattern 1 Register" on page 622	5b339c5dh
DCh	DFh	"Offset DCh: DDQSCADP0 - DQS Delay Calibration Aggressor Pattern 0 Register" on page 622	aaabffffh
E0h	E3h	"Offset E0h: DDQSCADP1 - DQS Delay Calibration Aggressor Pattern 1 Register" on page 623	db339ce1h
F0h	F3h	"Offset F0h: DIOMON - DDR I/O Monitor Register" on page 623	00000000h

Table 7-11. Bus 0, Device 0, Function 0: Summary of IMCH SMRBASE Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F8h	FBh	"Offset F8h: DRAMISCTL - Miscellaneous DRAM DDR Cluster Control Register" on page 624	1011h
C8h	CAh	"Offset C8h: DRAMDLLC - DDR I/O DLL Control Register" on page 625	0DB6C0h
E8h	EBh	"Offset E8h: FIVESREG - Fixed 5s Pattern Register" on page 625	55555555h
ECh	EFh	"Offset ECh: AAAAREG - Fixed A Pattern Register" on page 626	AAAAAAAAh
140h	143h	"Offset 140h: MBCSR - MemBIST Control Register" on page 626	00000000h
144h	147h	"Offset 144h: MBADDR - Memory Test Address Register" on page 629	00h
148h at 4h	14Ch at 4h	"Offset 148h: MBDATA[0:9] - Memory Test Data Register" on page 629	00h
19Ch	19Fh	"Offset 19Ch: MB_START_ADDR - Memory Test Start Address Register" on page 632	00h
1A0h	1A3h	"Offset 1A0h: MB_END_ADDR - Memory Test End Address Register" on page 632	00h
1A4h	1A7h	"Offset 1A4h: MBLFSRSED - Memory Test Circular Shift and LFSR Seed Register" on page 633	00h
1A8h	1ABh	"Offset 1A8h: MBFADDRPTR - Memory Test Failure Address Pointer Register" on page 633	00h
1B0h	1B3h	"Offset 1B0h: MB_ERR_DATA00 - Memory Test Error Data 0" on page 634	00h
1B4h	1B7h	"Offset 1B4h: MB_ERR_DATA01 - Memory Test Error Data 0" on page 634	00h
1B8h	1BBh	"Offset 1B8h: MB_ERR_DATA02 - Memory Test Error Data 0" on page 634	00h
1BCh	1BFh	"Offset 1BCh: MB_ERR_DATA03 - Memory Test Error Data 0" on page 635	00h
1C0h	1C1h	"Offset 1C0h: MB_ERR_DATA04 - Memory Test Error Data 0" on page 635	00h
1C4h	1C7h	"Offset 1C4h: MB_ERR_DATA10 - Memory Test Error Data 1" on page 635	00h
1C8h	1CBh	"Offset 1C8h: MB_ERR_DATA11 - Memory Test Error Data 1" on page 636	00h
1CCh	1CFh	"Offset 1CCh: MB_ERR_DATA12 - Memory Test Error Data 1" on page 636	00h
1D0h	1D3h	"Offset 1D0h: MB_ERR_DATA13 - Memory Test Error Data 1" on page 636	00h
1D4h	1D5h	"Offset 1D4h: MB_ERR_DATA14 - Memory Test Error Data 1" on page 637	00h
1D8h	1DBh	"Offset 1D8h: MB_ERR_DATA20 - Memory Test Error Data 2" on page 637	00h
1DCh	1DFh	"Offset 1DCh: MB_ERR_DATA21 - Memory Test Error Data 2" on page 637	00h
1E0h	1E3h	"Offset 1E0h: MB_ERR_DATA22 - Memory Test Error Data 2" on page 638	00h
1E4h	1E7h	"Offset 1E4h: MB_ERR_DATA23 - Memory Test Error Data 2" on page 638	00h
1E8h	1E9h	"Offset 1E8h: MB_ERR_DATA24 - Memory Test Error Data 2" on page 638	00h
1ECh	1EFh	"Offset 1ECh: MB_ERR_DATA30 - Memory Test Error Data 3" on page 639	00h
1F0h	1F4h	"Offset 1F0h: MB_ERR_DATA31 - Memory Test Error Data 3" on page 639	00h
1F4h	1F7h	"Offset 1F4h: MB_ERR_DATA32 - Memory Test Error Data 3" on page 639	00h
1F8h	1FBh	"Offset 1F8h: MB_ERR_DATA33 - Memory Test Error Data 3" on page 640	00h
1FCh	1FDh	"Offset 1FCh: MB_ERR_DATA34 - Memory Test Error Data 3" on page 640	00h
260h	263h	"Offset 260h: DDRIOMC0 - DDRIO Mode Register Control Register" on page 641	00000078h
264h	267h	"Offset 264h: DDRIOMC1 - DDRIO Mode Register Control Register 1" on page 642	52520000h
268h	26Bh	"Offset 268h: DDRIOMC2 - DDRIO Mode Control Register 2" on page 645	039E6000h
284h at 4h	294h at 4h	"Offset 284h: WL_CNTL[4:0] - Write Levelization Control Register" on page 647	00000000h
298h	29Bh	"Offset 298h: WDLL_MISC - DLL Miscellaneous Control" on page 649	00000000h

7.3.2 IMCH Error Reporting Registers: Bus 0, Device 0, Function 1

The IMCH includes the registers listed in Table 7-12. These registers materialize in PCI configuration space. See Section 16.2, "DRAM Controller Error Reporting Registers: Bus 0, Device 0, Function 1" for detailed discussion of these registers.

Table 7-12. Bus 0, Device 0, Function 1: Summary of IMCH Error Reporting PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 447	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 447	5021h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 448	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 448	0000h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 449	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 449	00h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 449	FFh
0Dh	0Dh	"Offset 0Dh: MLT - Master Latency Timer Register" on page 450	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 450	00h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 450	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 451	0000h
40h	43h	"Offset 40h: GLOBAL_FERR - Global First Error Register" on page 451	00000000h
44h	47h	"Offset 44h: GLOBAL_NERR - Global Next Error Register" on page 453	00000000h
48h	4Bh	"Offset 48h: NSI_FERR - NSI First Error Register" on page 454	00000000h
4Ch	4Fh	"Offset 4Ch: NSI_NERR - NSI Next Error Register" on page 457	00000000h
50h	53h	"Offset 50h: NSI_SCICMD - NSI SCI Command Register" on page 459	00000000h
54h	57h	"Offset 54h: NSI_SMICMD: NSI SMI Command Register" on page 461	00000000h
58h	5Bh	"Offset 58h: NSI_SERRCMD - NSI SERR Command Register" on page 464	00000000h
5Ch	5Fh	"Offset 5Ch: NSI_MCERRCMD - NSI MCERR Command Register" on page 466	00000000h
60h	61h	"Offset 60h: FSB_FERR - FSB First Error Register" on page 468	0000h
62h	63h	"Offset 62h: FSB_NERR - FSB Next Error Register" on page 469	0000h
64h	65h	"Offset 64h: FSB_EMASK - FSB Error Mask Register" on page 470	0009h
68h	69h	"offset 68h: FSB_SCICMD - FSB SCI Command Register" on page 471	0000h
6Ah	6Bh	"Offset 6Ah: FSB_SMICMD - FSB SMI Command Register" on page 472	0000h
6Ch	6Dh	"Offset 6Ch: FSB_SERRCMD - FSB SERR Command Register" on page 473	0000h
6Eh	6Fh	"Offset 6Eh: FSB_MCERRCMD - FSB MCERR Command Register" on page 474	0000h
70h	70h	"Offset 70h: BUF_FERR - Memory Buffer First Error Register" on page 475	00h
72h	72h	"Offset 72h: BUF_NERR - Memory Buffer Next Error Register" on page 475	00h
74h	74h	"Offset 74h: BUF_EMASK - Memory Buffer Error Mask Register" on page 476	00h
78h	78h	"Offset 78h: BUF_SCICMD - Memory Buffer SCI Command Register" on page 477	00h
7Ah	7Ah	"Offset 7Ah: BUF_SMICMD - Memory Buffer SMI Command Register" on page 478	00h
7Ch	7Ch	"Offset 7Ch: BUF_SERRCMD - Memory Buffer SERR Command Register" on page 479	00h

Table 7-12. Bus 0, Device 0, Function 1: Summary of IMCH Error Reporting PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
7Eh	7Eh	"Offset 7Eh: BUF_MCERRCMD - Memory Buffer MCERR Command Register" on page 480	00h
E4h	E7h	"Offset E4h: NSIERRINJCTL - NSI Error Injection Control Register" on page 481	00040000h
E8h	EBh	"Offset E8h: BERRINJCTL - Buffer Error Injection Control Register" on page 482	00000000h
80h	81h	"Offset 80h: DRAM_FERR - DRAM First Error Register" on page 483	0000h
82h	83h	"Offset 82h: DRAM_NERR - DRAM Next Error Register" on page 484	0000h
84h	84h	"Offset 84h: DRAM_EMASK - DRAM Error Mask Register" on page 486	00h
88h	88h	"Offset 88h: DRAM_SCICMD - DRAM SCI Command Register" on page 487	00h
8Ah	8Ah	"Offset 8Ah: DRAM_SMICMD - DRAM SMI Command Register" on page 488	00h
8Ch	8Ch	"Offset 8Ch: DRAM_SERRCMD - DRAM SERR Command Register" on page 489	00h
8Eh	8Eh	"Offset 8Eh: DRAM_MCERRCMD - DRAM MCERR Command Register" on page 490	00h
98h	99h	"Offset 98h: THRESH_SEC0 - Rank 0 SEC Error Threshold Register" on page 491	0000h
9Ah	9Bh	"Offset 9Ah: THRESH_SEC1 - Rank 1 SEC Error Threshold Register" on page 491	0000h
A0h	A3h	"Offset A0h: DRAM_SECF_ADD - DRAM First Single Bit Error Correct Address Register" on page 492	00000000h
A4h	A7h	"Offset A4h: DRAM_DED_ADD - DRAM Double Bit Error Address Register" on page 492	00000000h
A8h	ABh	"Offset A8h: DRAM_SCRB_ADD - DRAM Scrub Error Address Register" on page 493	00000000h
B0h	B1h	"Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register" on page 494	0000h
B2h	B3h	"Offset B2h: DRAM_DED_R0 - DRAM Rank 0 DED Error Counter Register" on page 494	0000h
B4h	B5h	"Offset B4h: DRAM_SEC_R1 - DRAM Rank 1 SEC Error Counter Register" on page 494	0000h
B6h	B7h	"Offset B6h: DRAM_DED_R1 - DRAM Rank 1 DED Error Counter Register" on page 495	0000h
C2h	C3h	"Offset C2h: THRESH_DED - DED Error Threshold Register" on page 495	0000h
C4h	C5h	"Offset C4h: DRAM_SECF_SYNDROME - DRAM First Single Error Correct Syndrome Register" on page 496	0000h
C6h	C7h	"Offset C6h: DRAM_SECN_SYNDROME - DRAM Next Single Error Correct Syndrome Register" on page 496	0000h
C8h	CBh	"Offset C8h: DRAM_SECN_ADD - DRAM Next Single Bit Error Correct Address Register" on page 497	00000000h
DCh	DDh	"Offset DCh: RANKTHREX - Rank Error Threshold Exceeded Register" on page 498	0000h
ECh	EFh	"Offset ECh: DERRINJCTL - DRAM Error Injection Control Register" on page 499	00000000h

7.3.3 EDMA Engine Registers: Bus 0, Device 1, Function 0

The EDMA engine includes the registers listed in Table 7-13 and Table 7-14. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 16.3, "EDMA Registers: Bus 0, Device 1, Function 0" and Section 16.6, "Memory Mapped I/O for EDMA Registers" for detailed discussion of these registers.

Table 7-13. Bus 0, Device 1, Function 0: Summary of EDMA PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 502	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 502	5023h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 503	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 504	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 504	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 505	80h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 505	08h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 505	00h
10h	13h	"Offset 10h: EDMALBAR - EDMA Low Base Address Register" on page 506	00000000h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 506	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 507	0000h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 507	B0h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 507	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 508	01h
40h	40h	"Offset 40h: EDMACTL - EDMA Control Register" on page 508	08h
80h	83h	"Offset 80h: EDMA_FERR - EDMA First Error Register" on page 509	00000000h
84h	87h	"Offset 84h: EDMA_NERR - EDMA Next Error Register" on page 511	00000000h
88h	88h	"Offset 88h: EDMA_EMASK - EDMA Error Mask Register" on page 513	00h
A0h	A0h	"Offset A0h: EDMA_SCICMD - EDMA SCI Command Register" on page 514	00h
A4h	A4h	"Offset A4h: EDMA_SMICMD - EDMA SMI Command Register" on page 515	00h
A8h	A8h	"Offset A8h: EDMA_SERRCMD - EDMA SERR Command Register" on page 516	00h
ACh	ACh	"Offset ACh: EDMA_MCERRCMD - EDMA MCERR Command Register" on page 517	00h
B0h	B3h	"Offset B0h: MSICR - MSI Control Register" on page 518	00020005h
B4h	B7h	"Offset B4h: MSIAR - MSI Address Register" on page 519	FEE00000h
B8h	B9h	"Offset B8h: MSIDR - MSI Data Register" on page 520	0000h

Table 7-14. Bus 0, Device 1, Function 0: Summary of EDMA Configuration Registers Mapped Through EDMALBAR Memory BAR (Sheet 1 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: CCR0 - Channel 0 Channel Control Register" on page 653	00000000h
04h	07h	"Offset 04h: CSR0 - Channel 0 Channel Status Register" on page 656	00000000h

Table 7-14. Bus 0, Device 1, Function 0: Summary of EDMA Configuration Registers Mapped Through EDMALBAR Memory BAR (Sheet 2 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
08h	0Bh	"Offset 08h: CDAR0 - Channel 0 Current Descriptor Address Register" on page 657	00000000h
0Ch	0Fh	"Offset 0Ch: CDUAR0 - Channel 0 Current Descriptor Upper Address Register" on page 658	00000000h
10h	13h	"Offset 10h: SAR0 - Channel 0 Source Address Register" on page 658	00000000h
14h	17h	"Offset 14h: SUAR0 - Channel 0 Source Upper Address Register" on page 659	00000000h
18h	1Bh	"Offset 18h: DAR0 - Channel 0 Destination Address Register" on page 659	00000000h
1Ch	1Fh	"Offset 1Ch: DUAR0 - Channel 0 Destination Upper Address Register" on page 660	00000000h
20h	23h	"Offset 20h: NDAR0 - Channel 0 Next Descriptor Address Register" on page 661	00000000h
24h	27h	"Offset 24h: NDUAR0 - Channel 0 Next Descriptor Upper Address Register" on page 662	00000000h
28h	2Bh	"Offset 28h: TCR0 - Channel 0 Transfer Count Register" on page 662	00000000h
2Ch	2Fh	"Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register" on page 663	00000000h
40h	43h	"Offset 40h: CCR1 - Channel 1 Channel Control Register" on page 665	00000000h
44h	47h	"Offset 44h: CSR1 - Channel 1 Channel Status Register" on page 665	00000000h
48h	4Bh	"Offset 48h: CDAR1 - Channel 1 Current Descriptor Address Register" on page 665	00000000h
4Ch	4Fh	"Offset 4Ch: CDUAR1 - Channel 1 Current Descriptor Upper Address Register" on page 666	00000000h
50h	53h	"Offset 50h: SAR1 - Channel 1 Source Address Register" on page 666	00000000h
54h	57h	"Offset 54h: SUAR1 - Channel 1 Source Upper Address Register" on page 666	00000000h
58h	5Bh	"Offset 58h: DAR1 - Channel 1 Destination Address Register" on page 667	00000000h
5Ch	5Fh	"Offset 5Ch: DUAR1 - Channel 1 Destination Upper Address Register" on page 667	00000000h
60h	63h	"Offset 60h: NDAR1 - Channel 1 Next Descriptor Address Register" on page 667	00000000h
64h	67h	"Offset 64h: NDUAR1 - Channel 1 Next Descriptor Upper Address Register" on page 668	00000000h
68h	6Bh	"Offset 68h: TCR1 - Channel 1 Transfer Count Register" on page 668	00000000h
6Ch	6Fh	"Offset 6Ch: DCR1 - Channel 1 Descriptor Control Register" on page 668	00000000h
80h	83h	"Offset 80h: CCR2 - Channel 2 Channel Control Register" on page 669	00000000h
84h	87h	"Offset 84h: CSR2 - Channel 2 Channel Status Register" on page 669	00000000h
88h	8Bh	"Offset 88h: CDAR2: Channel 2 Current Descriptor Address Register" on page 669	00000000h
8Ch	8Fh	"Offset 8Ch: CDUAR2 - Channel 2 Current Descriptor Upper Address Register" on page 670	00000000h
90h	93h	"Offset 90h: SAR2 - Channel 2 Source Address Register" on page 670	00000000h
94h	97h	"Offset 94h: SUAR2 - Channel 2 Source Upper Address Register" on page 670	00000000h
98h	9Bh	"Offset 98h: DAR2 - Channel 2 Destination Address Register" on page 671	00000000h
9Ch	9Fh	"Offset 9Ch: DUAR2 - Channel 2 Destination Upper Address Register" on page 671	00000000h
A0h	A3h	"Offset A0h: NDAR2 - Channel 2 Next Descriptor Address Register" on page 671	00000000h
A4h	A7h	"Offset A4h: NDUAR2 - Channel 2 Next Descriptor Upper Address Register" on page 672	00000000h
A8h	ABh	"Offset A8h: DCR2 - Channel 2Transfer Control Register" on page 672	00000000h
ACh	AFh	"Offset ACh: DCR2 - Channel 2 Descriptor Control Register" on page 672	00000000h
C0h	C3h	"Offset C0h: CCR3 - Channel 3 Channel Control Register" on page 673	00000000h
C4h	C7h	"Offset C4h: CSR3 - Channel 3 Channel Status Register" on page 673	00000000h

Table 7-14. Bus 0, Device 1, Function 0: Summary of EDMA Configuration Registers Mapped Through EDMALBAR Memory BAR (Sheet 3 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
C8h	CBh	"Offset C8h: CDAR3 - Channel 3 Current Descriptor Address Register" on page 673	00000000h
CCh	CFh	"Offset CCh: CDUAR3 - Channel 3 Current Descriptor Upper Address Register" on page 674	00000000h
D0h	D3h	"Offset D0h: SAR3 - Channel 3 Source Address Register" on page 674	00000000h
D4h	D7h	"Offset D4h: SUAR3 - Channel 3 Source Upper Address Register" on page 674	00000000h
D8h	DBh	"Offset D8h: DAR3 - Channel 3 Destination Address Register" on page 675	00000000h
DCh	DFh	"Offset DCh: DUAR3 - Channel 3 Destination Upper Address Register" on page 675	00000000h
E0h	E3h	"Offset E0h: NDAR3 - Channel 3 Next Descriptor Address Register" on page 675	00000000h
E4h	E7h	"Offset E4h: NDUAR3 - Channel 3 Next Descriptor Upper Address Register" on page 676	00000000h
E8h	EBh	"Offset E8h: TCR3 - Channel 3 Transfer Count Register" on page 676	00000000h
ECh	EFh	"Offset ECh: DCR3 - Channel 3 Descriptor Control Register" on page 677	00000000h
100h	103h	"Offset 100h: DCGC - EDMA Controller Global Command" on page 677	00000000h
104h	107h	"Offset 104h: DCGS - EDMA Controller Global Status" on page 678	00000000h

7.3.4 PCI Express* Port A Registers: Bus 0, Device 2, Function 0

The PCI Express* Port A includes the registers listed in Table 7-15. These registers materialize in PCI configuration spaces. See Section 16.4, "PCI Express* Port A Standard and Enhanced Registers: Bus 0, Devices 2 and 3, Function 0" for detailed discussion of these registers.

Table 7-15. Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 1 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 527	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 527	5024h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 528	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 530	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 531	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 532	04h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 532	06h
0Ch	0Ch	"Offset 0Ch: CLS - Cache Line Size Register" on page 533	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 533	01h
18h	18h	"Offset 18h: PBUSN - Primary Bus Number Register" on page 534	00h
19h	19h	"Offset 19h: SBUSN - Secondary Bus Number Register" on page 534	00h
1Ah	1Ah	"Offset 1Ah: SUBUSN: Subordinate Bus Number Register" on page 535	00h
1Ch	1Ch	"Offset 1Ch: IOBASE - I/O Base Address Register" on page 535	F0h
1Dh	1Dh	"Offset 1Dh: IOLIMIT - I/O Limit Address Register" on page 536	00h
1Eh	1Fh	"Offset 1Eh: SECSTS - Secondary Status Register" on page 536	0000h
20h	21h	"Offset 20h: MBASE - Memory Base Address Register" on page 538	FFF0h
22h	23h	"Offset 22h: MLIMIT - Memory Limit Address Register" on page 539	0000h
24h	25h	"Offset 24h: PMBASE - Prefetchable Memory Base Address Register" on page 540	FFF1h
26h	27h	"Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register" on page 540	0001h
28h	28h	"Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register" on page 541	0Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register" on page 541	00h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 542	50h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 542	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 543	01h
3Eh	3Eh	"Offset 3Eh: BCTRL - Bridge Control Register" on page 543	00h
44h	44h	"Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register" on page 545	00h
45h	45h	"Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register" on page 546	00h
46h	46h	"Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register" on page 547	00h
47h	47h	"Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register" on page 547	00h
48h	48h	"Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register" on page 548	00h
50h	50h	"Offset 50h: PMCAPID - Power Management Capabilities Structure Register" on page 548	01h

Table 7-15. Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 2 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
51h	51h	"Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register" on page 549	58h
52h	53h	"Offset 52h: PMCAPA - Power Management Capabilities Register" on page 549	C822h
54h	55h	"Offset 54h: PMCSR - Power Management Status and Control Register" on page 550	0000h
56h	56h	"Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register" on page 551	00h
58h	58h	"Offset 58h: MSICAPID - MSI Capabilities Structure Register" on page 551	05h
59h	59h	"Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register" on page 552	64h
5Ah	5Bh	"Offset 5Ah: MSICAPA - MSI Capabilities Register" on page 553	0002h
5Ch	5Fh	"Offset 5Ch: MSIAR - MSI Address for PCI Express Register" on page 553	FEE00000h
60h	61h	"Offset 60h: MSIDR - MSI Data Register" on page 554	0000h
64h	64h	"Offset 64h: PEACAPID - PCI Express Features Capabilities ID Register" on page 555	10h
65h	65h	"Offset 65h: PEANPTR - PCI Express Next Capabilities Pointer Register" on page 556	00h
66h	67h	"Offset 66h: PEACAPA - PCI Express Features Capabilities Register" on page 556	0041h
68h	6Bh	"Offset 68h: PEADEVCAP - PCI Express Device Capabilities Register" on page 557	00000001h
6Ch	6Dh	"Offset 6Ch: PEADEVCTL - PCI Express Device Control Register" on page 558	0000h
6Eh	6Fh	"Offset 6Eh: PEADEVSTS - PCI Express Device Status Register" on page 560	0000h
70h	73h	"Offset 70h: PEALNKCAP - PCI Express Link Capabilities Register" on page 561	0203E481h
74h	75h	"Offset 74h: PEALNKCTL - PCI Express Link Control Register" on page 562	0001h
76h	77h	"Offset 76h: PEALNKSTS - PCI Express Link Status Register" on page 564	1001h
78h	7Bh	"Offset 78h: PEASLTCAP - PCI Express Slot Capabilities Register" on page 565	00000000h
7Ch	7Dh	"Offset 7Ch: PEASLTCTL - PCI Express Slot Control Register" on page 568	01C0h
7Eh	7Fh	"Offset 7Eh: PEASLTSTS - PCI Express Slot Status Register" on page 569	0040h
80h	83h	"Offset 80h: PEARPCTL - PCI Express Root Port Control Register" on page 570	00000000h
84h	87h	"Offset 84h: PEARPSTS - PCI Express Root Port Status Register" on page 571	00000000h
100h	103h	"Offset 100h: ENHCAPST - Enhanced Capability Structure Register" on page 571	00010001h
104h	107h	"Offset 104h: UNCERRSTS - Uncorrectable Error Status Register" on page 572	00000000h
108h	10Bh	"Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register" on page 574	00000000h
10Ch	10Fh	"Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register" on page 575	00062010h
110h	113h	"Offset 110h: CORERRSTS - Correctable Error Status Register" on page 576	00000000h
114h	117h	"Offset 114h: CORERRMSK - Correctable Error Mask Register" on page 578	00000000h
118h	11Bh	"Offset 118h: AERCACR - Advanced Error Capabilities and Control Register" on page 579	00000000h
11Ch	11Fh	"Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register" on page 580	00000000h
120h	123h	"Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register" on page 580	00000000h
124h	127h	"Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register" on page 581	00000000h
128h	12Bh	"Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register" on page 581	00000000h
12Ch	12Fh	"Offset 12Ch: RPERRCMD - Root (Port) Error Command Register" on page 582	00000000h

Table 7-15. Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 3 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
130h	133h	"Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register" on page 583	00000000h
134h	137h	"Offset 134h: ERRSID - Error Source ID Register" on page 585	00000000h
140h	143h	"Offset 140h: PEAUNITERR - PCI Express Unit Error Register" on page 586	00000000h
144h	147h	"Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register" on page 588	0000E000h
148h	14Bh	"Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register" on page 589	00000000h
14Ch	14Fh	"Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register" on page 591	00000000h
150h	153h	"Offset 150h: COREDMASK - Correctable Error Detect Mask Register" on page 592	00000000h
158h	15Bh	"Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register" on page 594	00000000h
160h	163h	"Offset 160h: PEAFERR - PCI Express First Error Register" on page 595	00000000h
164h	167h	"Offset 164h: PEANERR - PCI Express Next Error Register" on page 597	00000000h
168h	16Bh	"Offset 168h: PEAERRINJCTL - Error Injection Control Register" on page 597	00000000h

7.3.5 PCI Express* Port A1 Registers: Bus 0, Device 3, Function 0

The PCI Express* Port A1 includes the registers listed in Table 7-16. These registers materialize in PCI configuration space. See Section 16.4, "PCI Express* Port A Standard and Enhanced Registers: Bus 0, Devices 2 and 3, Function 0" for detailed discussion of these registers.

Table 7-16. Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 1 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 527	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 528	5025h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 528	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 530	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 531	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 532	04h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 532	06h
0Ch	0Ch	"Offset 0Ch: CLS - Cache Line Size Register" on page 533	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 533	01h
18h	18h	"Offset 18h: PBUSN - Primary Bus Number Register" on page 534	00h
19h	19h	"Offset 19h: SBUSN - Secondary Bus Number Register" on page 534	00h
1Ah	1Ah	"Offset 1Ah: SUBUSN: Subordinate Bus Number Register" on page 535	00h
1Ch	1Ch	"Offset 1Ch: IOBASE - I/O Base Address Register" on page 535	F0h
1Dh	1Dh	"Offset 1Dh: IOLIMIT - I/O Limit Address Register" on page 536	00h
1Eh	1Fh	"Offset 1Eh: SECSTS - Secondary Status Register" on page 536	0000h
20h	21h	"Offset 20h: MBASE - Memory Base Address Register" on page 538	FFF0h
22h	23h	"Offset 22h: MLIMIT - Memory Limit Address Register" on page 539	0000h
24h	25h	"Offset 24h: PMBASE - Prefetchable Memory Base Address Register" on page 540	FFF1h
26h	27h	"Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register" on page 540	0001h
28h	28h	"Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register" on page 541	0Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register" on page 541	00h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 542	50h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 542	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 543	01h
3Eh	3Eh	"Offset 3Eh: BCTRL - Bridge Control Register" on page 543	00h
44h	44h	"Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register" on page 545	00h
45h	45h	"Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register" on page 546	00h
46h	46h	"Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register" on page 547	00h
47h	47h	"Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register" on page 547	00h
48h	48h	"Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register" on page 548	00h
50h	50h	"Offset 50h: PMCAPID - Power Management Capabilities Structure Register" on page 548	01h

Table 7-16. Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 2 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
51h	51h	"Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register" on page 549	58h
52h	53h	"Offset 52h: PMCAPA - Power Management Capabilities Register" on page 549	C822h
54h	55h	"Offset 54h: PMCSR - Power Management Status and Control Register" on page 550	0000h
56h	56h	"Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register" on page 551	00h
58h	58h	"Offset 58h: MSICAPID - MSI Capabilities Structure Register" on page 551	05h
59h	59h	"Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register" on page 552	64h
5Ah	5Bh	"Offset 5Ah: MSICAPA - MSI Capabilities Register" on page 553	0002h
5Ch	5Fh	"Offset 5Ch: MSIAR - MSI Address for PCI Express Register" on page 553	FEE00000h
60h	61h	"Offset 60h: MSIDR - MSI Data Register" on page 554	0000h
64h	64h	"Offset 64h: PEACAPID - PCI Express Features Capabilities ID Register" on page 555	10h
65h	65h	"Offset 65h: PEANPTR - PCI Express Next Capabilities Pointer Register" on page 556	00h
66h	67h	"Offset 66h: PEACAPA - PCI Express Features Capabilities Register" on page 556	0041h
68h	6Bh	"Offset 68h: PEADEVCAP - PCI Express Device Capabilities Register" on page 557	00000001h
6Ch	6Dh	"Offset 6Ch: PEADEVCTL - PCI Express Device Control Register" on page 558	0000h
6Eh	6Fh	"Offset 6Eh: PEADEVSTS - PCI Express Device Status Register" on page 560	0000h
70h	73h	"Offset 70h: PEA1LNKCAP - PCI Express Link Capabilities Register" on page 561	0303E441h
74h	75h	"Offset 74h: PEALNKCTL - PCI Express Link Control Register" on page 562	0001h
76h	77h	"Offset 76h: PEALNKSTS - PCI Express Link Status Register" on page 564	1001h
78h	7Bh	"Offset 78h: PEA1SLTCAP - PCI Express Slot Capabilities Register" on page 566	00000000h
7Ch	7Dh	"Offset 7Ch: PEASLTCTL - PCI Express Slot Control Register" on page 568	01C0h
7Eh	7Fh	"Offset 7Eh: PEASLTSTS - PCI Express Slot Status Register" on page 569	0040h
80h	83h	"Offset 80h: PEARPCTL - PCI Express Root Port Control Register" on page 570	00000000h
84h	87h	"Offset 84h: PEARPSTS - PCI Express Root Port Status Register" on page 571	00000000h
100h	103h	"Offset 100h: ENHCAPST - Enhanced Capability Structure Register" on page 571	00010001h
104h	107h	"Offset 104h: UNCERRSTS - Uncorrectable Error Status Register" on page 572	00000000h
108h	10Bh	"Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register" on page 574	00000000h
10Ch	10Fh	"Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register" on page 575	00062010h
110h	113h	"Offset 110h: CORERRSTS - Correctable Error Status Register" on page 576	00000000h
114h	117h	"Offset 114h: CORERRMSK - Correctable Error Mask Register" on page 578	00000000h
118h	11Bh	"Offset 118h: AERCACR - Advanced Error Capabilities and Control Register" on page 579	00000000h
11Ch	11Fh	"Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register" on page 580	00000000h
120h	123h	"Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register" on page 580	00000000h
124h	127h	"Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register" on page 581	00000000h
128h	12Bh	"Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register" on page 581	00000000h
12Ch	12Fh	"Offset 12Ch: RPERRCMD - Root (Port) Error Command Register" on page 582	00000000h

Table 7-16. Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 3 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
130h	133h	"Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register" on page 583	00000000h
134h	137h	"Offset 134h: ERRSID - Error Source ID Register" on page 585	00000000h
140h	143h	"Offset 140h: PEAUNITERR - PCI Express Unit Error Register" on page 586	00000000h
144h	147h	"Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register" on page 588	0000E000h
148h	14Bh	"Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register" on page 589	00000000h
14Ch	14Fh	"Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register" on page 591	00000000h
150h	153h	"Offset 150h: COREDMASK - Correctable Error Detect Mask Register" on page 592	00000000h
158h	15Bh	"Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register" on page 594	00000000h
160h	163h	"Offset 160h: PEAFERR - PCI Express First Error Register" on page 595	00000000h
164h	167h	"Offset 164h: PEANERR - PCI Express Next Error Register" on page 597	00000000h
168h	16Bh	"Offset 168h: PEAERRINJCTL - Error Injection Control Register" on page 597	00000000h

7.3.6 USB (1.1) Controller: Bus 0, Device 29, Functions 0

The USB 1.1 controller includes the registers listed in Table 7-17 and Table 7-18. These registers materialize in PCI configuration and I/O spaces (via PCI I/O BAR), respectively. See Chapter 25.0, "USB (1.1) Controller: Bus 0, Device 29, Function 0" for detailed discussion of these registers.

Table 7-17. Bus 0, Device 29, Functions 0, Summary of USB (1.1) Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"ID - Identifiers Register" on page 942	50338086h
04h	05h	"PCICMD - Command Register" on page 942	0000h
06h	07h	"PCISTS - Device Status Register" on page 943	0280h
08h	08h	"RID - Revision ID Register" on page 944	Variable
0Ah	0Ah	"SUBC - Sub Class Code Register" on page 945	03h
0Bh	0Bh	"BCC - Base Class Code Register" on page 945	0Ch
0Dh	0Dh	"MLT - Master Latency Timer Register" on page 945	00h
0Eh	0Eh	"HDR - Header Type Register" on page 946	Variable
20h	23h	"USBIOBAR - Base Address Register" on page 946	00000001h
2Ch	2Dh	"USBx_SVID - USB Subsystem Vendor ID Register" on page 947	0000h
2Eh	2Fh	"USBx_SID - USB Subsystem ID Register" on page 947	0000h
3Ch	3Ch	"INTL - Interrupt Line Register" on page 948	00h
3Dh	3Dh	"INTP - Interrupt Pin Register" on page 948	Variable
60h	60h	"SBRN - Serial Bus Release Number Register" on page 948	10h
C0h	C1h	"USBLKMCR - USB Legacy Keyboard/Mouse Control Register" on page 949	2000h
C4h	C4h	"USBREN - USB Resume Enable Register" on page 951	00h
C8h	C8h	"USBCWP - USB Core Well Policy Register" on page 951	00h
F8h	FBh	"MANID - Manufacturer ID Register" on page 952	00010F90h

Table 7-18. Summary of USB (1.1) Controller Configuration Registers Mapped Through USBIOBAR I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"USBCMD: USB Command Register" on page 954	0000h
02h	03h	"USBSTS: USB Status Register" on page 957	0020h
04h	05h	"USBINTR: USB Interrupt Enable Register" on page 959	0000h
06h	07h	"FRNUM: Frame Number Register" on page 959	0000h
08h	0Bh	"FRBASEADD: Frame List Base Address Register" on page 960	XXXXX000h
0Ch	0Ch	"SOFMOD: Start of Frame Modify Register" on page 961	40h
10h	11h	"PSCR - Port Status and Control Register" on page 962	0080h
12h	13h	"PSCR - Port Status and Control Register" on page 962	0080h

7.3.7 USB (2.0) Controller: Bus 0, Device 29, Function 7

The USB 2.0 controller includes the registers listed in Table 7-19 and Table 7-20. These registers materialize in PCI configuration and I/O spaces (via PCI I/O BAR), respectively. See Chapter 26.0, "USB 2.0 PCI Configuration Registers" for detailed discussion of these registers.

Table 7-19. Bus 0, Device 29, Function 7: Summary of USB (2.0) Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor ID Register" on page 979	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 979	5035h
04h	05h	"Offset 04h: CMD - Command Register" on page 980	0000h
06h	07h	"Offset 06h: DSR - Device Status Register" on page 981	0290h
08h	08h	"Offset 08h: RID - Revision ID Register" on page 983	Variable
09h	09h	"Offset 09h: PI - Programming Interface Register" on page 983	20h
0Ah	0Ah	"Offset 0Ah: SCC - Sub Class Code Register" on page 983	03h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 984	0Ch
0Dh	0Dh	"Offset 0Dh: MLT - Master Latency Timer Register" on page 984	00h
10h	13h	"Offset 10h: MBAR - Memory Base Address Register" on page 985	00000000h
2Ch	2Dh	"Offset 2Ch: SSVID - USB 2.0 Subsystem Vendor ID Register" on page 985	XXXXh
2Eh	2Fh	"Offset 2Eh: SSID - USB 2.0 Subsystem ID Register" on page 986	XXXXh
34h	34h	"Offset 34h: CAP_PTR - Capabilities Pointer Register" on page 986	50h
3Ch	3Ch	"Offset 3Ch: ILINE - Interrupt Line Register" on page 987	00h
3Dh	3Dh	"Offset 3Dh: IPIN - Interrupt Pin Register" on page 987	Variable
50h	50h	"Offset 50h: PM_CID - PCI Power Management Capability ID Register" on page 987	01h
51h	51h	"Offset 51h: PM_NEXT - Next Item Pointer #1 Register" on page 988	58h
52h	53h	"Offset 52h: PM_CAP - Power Management Capabilities Register" on page 989	C9C2h
54h	55h	"Offset 54h: PM_CS - Power Management Control/Status Register" on page 990	0000h
58h	58h	"Offset 58h: DP_CID - Debug Port Capability ID Register" on page 991	0Ah
59h	59h	"Offset 59h: DP_NEXT - Next Item Pointer #2 Register" on page 991	00h
5Ah	5Bh	"Offset 5Ah: DP_BASE - Debug Port Base Offset Register" on page 991	20A0h
60h	60h	"Offset 60h: SBRN - Serial Bus Release Number Register" on page 992	20h
61h	61h	"Offset 61h: FLA - Frame Length Adjustment Register" on page 992	20h
62h	63h	"Offset 62h: PWC - Port Wake Capability Register" on page 993	01FFh
64h	65h	"Offset 64h: CUO - Classic USB Override Register" on page 994	0000h
68h	6Bh	"Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register" on page 994	00000001h
6Ch	6Fh	"Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register" on page 995	00000000h
70h	73h	"Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register" on page 997	00000000h
80h	80h	"Offset 80h: AC - Access Control Register" on page 999	00h
F8h	FBh	"Offset F8h: MANID - Manufacturer ID Register" on page 1000	00010F90h

Table 7-20. Bus 0, Device 29, Function 7: Summary of USB (2.0) Controller Configuration Registers Mapped Through MBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: CAPLENGTH - Capability Length Register" on page 1002	20h
02h	03h	"Offset 02h: HCIVERSION - Host Controller Interface Version Number Register" on page 1003	0100h
04h	07h	"Offset 04h: HCSPARAMS - Host Controller Structural Parameters Register" on page 1003	01001202h
08h	0Bh	"Offset 08h: HCCPARAMS - Host Controller Capability Parameters Register" on page 1004	00006871h
20h	23h	"Offset 20h: USB2CMD - USB 2.0 Command Register" on page 1007	00080000h
24h	27h	"Offset 24h: USB2STS - USB 2.0 Status Register" on page 1009	00001000h
28h	2Bh	"Offset 28h: USB2INTR - USB 2.0 Interrupt Enable Register" on page 1012	00000000h
2Ch	2Fh	"Offset 2Ch: FRINDEX - Frame Index Register" on page 1013	00000000h
30h	33h	"Offset 30h: CTRLDSSEGMENT - Control Data Structure Segment Register" on page 1014	00000000h
34h	37h	"Offset 34h: PERIODICLISTBASE - Periodic Frame List Base Address Register" on page 1014	00000XXXh
38h	3Bh	"Offset 38h: ASYNCLISTADDR - Current Asynchronous List Address Register" on page 1015	00000000h
60h	63h	"Offset 60h: CONFIGFLAG - Configure Flag Register" on page 1015	00000000h
64h	67h	"Offset 64h: PORTSC - Port N Status and Control Register" on page 1016	00003000h
68h	6Bh	"Offset 64h: PORTSC - Port N Status and Control Register" on page 1016	00003000h
A0h	A3h	"Offset A0h: CNTL_STS - Control/Status Register" on page 1037	00000000h
A4h	A4h	"Offset A4h: USBPID - USB PIDs Register" on page 1039	00000000h
A8h	AFh	"Offset A8h: DATABUF - Data Buffer Bytes 7:0" on page 1039	0000000000000000h
B0h	B0h	"Offset B0h: CONFIG - Configuration Register" on page 1040	00007F01h

7.3.8 Root Complex: Bus 0, Device 31, Function 0

The Root Complex includes the registers listed in Table 7-21. These registers materialize in memory space (via PCI memory BAR), respectively. See Chapter 17.0, "Bridging and Configuration" for detailed discussion of these registers.

Table 7-21. Bus 0, Device 31, Function 0: Summary of Root Complex Configuration Registers Mapped Through RCBA Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: VCH - Virtual Channel Capability Header Register" on page 691	10010002h
0004h	0007h	"Offset 0004h: VCAP1 - Virtual Channel Capability 1 Register" on page 691	0801h
0008h	000Bh	"Offset 0008h: VCAP2 - Virtual Channel Capability 2 Register" on page 692	0001h
000Ch	000Dh	"Offset 000Ch: PVC - Port Virtual Channel Control Register" on page 692	0h
000Eh	000Fh	"Offset 000Eh: PVS -Port Virtual Channel Status Register" on page 693	0h
0010h	0013h	"Offset 0010h: V0CAP - Virtual Channel 0 Resource Capability Register" on page 693	00000001h
0014h	0017h	"Offset 0014h: V0CTL - Virtual Channel 0 Resource Control Register" on page 694	800000FFh
001Ah	001Bh	"Offset 001Ah: V0STS - Virtual Channel 0 Resource Status Register" on page 695	0h
0100h	0103h	"Offset 0100h: RCTCL - Root Complex Topology Capabilities List Register" on page 696	1A010005h
0104h	0107h	"Offset 0104h: ESD - Element Self Description Register" on page 696	00000102h
0110h	0113h	"Offset 0110h: ULD - Upstream Link Description Register" on page 697	0001h
0118h	011Fh	"Offset 0118h: ULBA - Upstream Link Base Address Register" on page 697	0000000000000000h
01A0h	01A3h	"Offset 01A0h: ILCL - Internal Link Capabilities List Register" on page 698	00010006h
01A4h	01A7h	"Offset 01A4h: LCAP - Link Capabilities Register" on page 698	0012441h
01A8h	01A9h	"Offset 01A8h: LCTL - Link Control Register" on page 699	0h
01AAh	01ABh	"Offset 01AAh: LSTS - Link Status Register" on page 700	0041h
3108h	310Bh	"Offset 3108h: D29IP - Device 29 Interrupt Pin Register" on page 702	10004321h
3140h	3141h	"Offset 3140h: D31IR - Device 31 Interrupt Route Register" on page 702	3210h
3144h	3145h	"Offset 3144h: D29IR - Device 29 Interrupt Route Register" on page 703	3210h
31FFh	31FFh	"Offset 31FFh: OIC - Other Interrupt Control Register" on page 704	0h
3400h	3403h	"Offset 3400h: RC - RTC Configuration Register" on page 704	0h
3404h	3407h	"Offset 3404h: HPTC - High Performance Precision Timer Configuration Register" on page 705	0h
3410h	3413h	"Offset 3410h: GCS - General Control and Status Register" on page 706	Variable
3414h	3417h	"Offset 3414h: BUC - Backed Up Control Register" on page 708	Variable
3418h	341Bh	"Offset 3418h: FD - Function Disable Register" on page 709	00000080h
341Ch	341Fh	"Offset 341Ch: PRC - Power Reduction Control Register Clock Gating" on page 711	0h

7.3.9 LPC Interface: Bus 0, Device 31, Function 0

The LPC interface includes the registers listed in Table 7-22 through Table 7-26. These registers materialize in PCI configuration and I/O spaces (via PCI I/O BAR). See Chapter 19.0, "LPC Interface: Bus 0, Device 31, Function 0", Chapter 27.0, "Power Management", Chapter 18.0, "System Management", and Chapter 22.0, "General Purpose I/O: Bus 0, Device 31, Function 0" for detailed discussion of these registers.

Table 7-22. Bus 0, Device 31, Function 0: Summary of LPC Interface PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: ID: Vendor Identification Register" on page 734	50318086h
04h	05h	"Offset 04h: CMD: Device Command Register" on page 735	0007h
06h	07h	"Offset 06h: STS: Status Register" on page 736	0200h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 737	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 737	060100h
0Dh	0Dh	"Offset 0Dh: MLT: Master Latency Timer Register" on page 737	00h
0Eh	0Eh	"Offset 0Eh: HTYPE: Header Type Register" on page 738	80H
2Ch	2Fh	"Offset 2Ch: SID: Subsystem Identifiers Register" on page 738	00000000h
40h	43h	"Offset 40h: ABASE: ACPI Base Address Register" on page 739	00000001h
44h	47h	"Offset 44h: ACTL: ACPI Control Register" on page 739	00h
48h	48h	"Offset 48h: GBA: GPIO Base Address Register" on page 740	00000001h
4Ch	4Ch	"Offset 4Ch: GC: GPIO Control Register" on page 741	00h
60h	60h	"Offset 60h: PARC: PIRQA Routing Control Register" on page 741	80h
61h	61h	"Offset 61h: PBRC: PIRQB Routing Control Register" on page 742	80h
62h	62h	"Offset 62h: PCRC: PIRQC Routing Control Register" on page 742	80h
63h	63h	"Offset 63h: PDRC: PIRQDQ Routing Control Register" on page 743	80h
64h	64h	"Offset 64h: SCNT: Serial IRQ Control Register" on page 744	10h
68h	68h	"Offset 68h: PERC: PIRQEQ Routing Control Register" on page 745	80h
69h	69h	"Offset 69h: PFRC: PIRQF Routing Control Register" on page 745	80h
6Ah	6Ah	"Offset 6Ah: PGRC: PIRQG Routing Control Register" on page 746	80h
6Bh	6Bh	"Offset 6Bh: PHRC: PIRQH Routing Control Register" on page 747	80h
80h	81h	"Offset 80h: IOD: I/O Decode Ranges Register" on page 747	0000h
82h	83h	"Offset 82h: IOE: I/O Enables Register" on page 749	0000h
84h	85h	"Offset 84h: LG1: LPC Generic Decode Range 1 Register" on page 750	0000h
88h	88h	"Offset 88h: LG2: LPC Generic Decode Range 2 Register" on page 751	0000h
D0h	D3h	"Offset D0h: FS1: FWH ID Select 1 Register" on page 752	00112233h
D4h	D5h	"Offset D4h: FS2: FWH ID Select 2 Register" on page 753	4567h
D8h	DBh	"Offset D8h: FDE: FWH Decode Enable Register" on page 754	FFCFh
DCh	DCh	"Offset DCh: BC: BIOS Control Register" on page 756	00h
F0h	F3h	"Offset F0h: RCBA: Root Complex Base Address Register" on page 757	00000000h
F8h	FBh	"Offset F8h: MANID: Manufacturer ID Register" on page 757	00010F90h

Table 7-23. Bus 0, Device 31, Function 0: Summary of LPC Interface Power Management PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
A0h	A0h	"Offset A0h: GEN_PMCON_1 - General PM Configuration 1 Register" on page 1048	0200h
A2h	A2h	"Offset A2h: GEN_PMCON_2 - General PM Configuration 2 Register" on page 1049	00h
A4h	A4h	"Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register" on page 1051	00h
B8h	BBh	"Offset B8h: GPI_ROUT - GPI Routing Control Register" on page 1053	00000000h

Table 7-24. Bus 0, Device 31, Function 0: Summary of TCO Configuration Registers Mapped Through TCOBASE I/O BAR"

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: TRLD - TCO Timer Reload and Current Value Register" on page 715	0000h
02h	02h	"Offset 02h: TDI - TCO Data In Register" on page 715	00h
03h	03h	"Offset 03h: TDO - TCO Data Out Register" on page 716	00h
04h	04h	"Offset 04h: TSTS1 - TCO 1 Status Register" on page 716	0000h
06h	07h	"Offset 06h: TSTS2 - TCO 2 STS Register" on page 718	0000h
08h	09h	"Offset 08h: TCTL1 - TCO 1 Control Register" on page 720	0000h
0Ah	0Bh	"Offset 0Ah: TCTL2 - TCO 2 Control Register" on page 721	0008h
0Ch at 01h	0Ch at 01h	"Offset 0Ch: TMSG[1-2] - TCO MESSAGE Register" on page 721	00h
0Eh	0Eh	"Offset 0Eh: TWDS - TCO Watchdog Status Register" on page 722	00h
10h	10h	"Offset 10h: LE - Legacy Elimination Register" on page 722	03h
12h	13h	"Offset 12h: TTMR - TCO Timer Initial Value Register" on page 723	0004h

Table 7-25. Bus 0, Device 31, Function 0: Summary of LPC Interface Power Management General Configuration Registers Mapped Through PMBASE I/O BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PM1_STS – Power Management 1 Status Register" on page 1056	0000h
02h	02h	"Offset 02h: PM1_EN - Power Management 1 Enables Register" on page 1058	0000h
04h	04h	"Offset 04h: PM1_CNT - Power Management 1 Control Register" on page 1059	0000h
08h	B8h	"Offset 08h: PM1_TMR - Power Management 1 Timer Register" on page 1060	00000000h
10h	10h	"Offset 10h: PROC_CNT - Processor Control Register" on page 1060	00000000h
14h	14h	"Offset 14h: LV2 - Level 2 Register" on page 1063	00h
28h	28h	"Offset 28h: GPE0_STS - General Purpose Event 0 Status Register" on page 1063	00000000h
2Ch	2Ch	"Offset 2Ch: PMBASE_GPE0_EN - General Purpose Event 0 Enables Register" on page 1067	00000000h

Table 7-25. Bus 0, Device 31, Function 0: Summary of LPC Interface Power Management General Configuration Registers Mapped Through PMBASE I/O BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
30h	30h	"Offset 30h: SMI_EN - SMI Control and Enable Register" on page 1068	00000000h
34h	34h	"Offset 34h: SMI_STS - SMI Status Register" on page 1071	00000000h
38h	38h	"Offset 38h: ALT_GPI_SMI_EN - Alternate GPI SMI Enable Register" on page 1073	0000h
3Ah	3Ah	"Offset 3Ah: ALT_GPI_SMI_STS - Alternate GPI SMI Status Register" on page 1074	0000h
44h	44h	"Offset 44h: DEVTRAP_STS - DEVTRAP_STS Register" on page 1074	0000h

Table 7-26. Bus 0, Device 31, Function 0: Summary of General Purpose I/O Configuration Registers Mapped Through GBA BAR IO BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: GPIO_USE_SEL1 - GPIO Use Select 1 {31:0} Register" on page 807	Variable
04h	07h	"Offset 04h: GP_IO_SEL1 - GPIO Input/Output Select 1 {31:0} Register" on page 808	E400FFFFh
0Ch	0Fh	"Offset 0Ch: GP_LVL1 - GPIO Level 1 for Input or Output {31:0} Register" on page 809	FF3F0000h
18h	1Bh	"Offset 18h: GPO_BLINK - GPIO Blink Enable Register" on page 810	00040000h
2Ch	2Fh	"Offset 2Ch: GPI_INV - GPIO Signal Invert Register" on page 812	00000000h
30h	33h	"Offset 30h: GPIO_USE_SEL2 - GPIO Use Select 2 {63:32} Register" on page 813	Variable
34h	37h	"Offset 34h: GP_IO_SEL2 - GPIO Input/Output Select 2 {63:32} Register" on page 813	00000300h
38h	3Bh	"Offset 38h: GP_LVL2 - GPIO Level for Input or Output 2 {63:32} Register" on page 814	00030207h

7.3.10 SATA Controller: Bus 0, Device 31, Function 2

The SATA controller includes the registers listed in Table 7-27, Table 7-28, and Table 7-29. These registers materialize in PCI configuration, I/O, and memory spaces (via PCI I/O and memory BARs). See Chapter 23.0, "SATA: Bus 0, Device 31, Function 2" for detailed discussion of these registers.

Table 7-27. Bus 0, Device 31, Function 2: Summary of SATA Controller PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: ID – Identifiers Register" on page 819	Variable
04h	05h	"Offset 04h: CMD - Command Register" on page 819	0000h
06h	07h	"Offset 06h: STS - Device Status Register" on page 820	02B0h
08h	08h	"Offset 08h: RID - Revision ID Register" on page 821	Variable
0Ah	0Bh	"Offset 0Ah: CC - Class Code Register" on page 823	Variable
0Dh	0Dh	"Offset 0Dh: MLT – Master Latency Timer Register" on page 823	00h
10h	13h	"Offset 10h: PCMDBA – Primary Command Block Base Address Register" on page 824	00000001h
14h	17h	"Offset 14h: PCTLBA – Primary Control Block Base Address Register" on page 824	00000001h
18h	1Bh	"Offset 18h: SCMDBA – Secondary Command Block Base Address Register" on page 825	00000001h
1Ch	1Fh	"Offset 1Ch: SCTLBA – Secondary Control Block Base Address Register" on page 825	00000001h
20h	23h	"Offset 20h: LBAR – Legacy Bus Master Base Address Register when SCC is SATA with AHCI PI" on page 826	00000001h
24h	27h	"Offset 24h: ABAR – AHCI Base Address Register" on page 826	00000000h
2Ch	2Fh	"Offset 2Ch: SS - Sub System Identifiers Register" on page 827	00000000h
34h	34h	"Offset 34h: CAP – Capabilities Pointer Register" on page 827	80h
3Ch	3Dh	"Offset 3Ch: INTR - Interrupt Information Register" on page 828	Variable
40h	41h	"Offset 40h: PTIM – Primary Timing Register" on page 829	0000h
44h	44h	"Offset 44h: D1TIM – Device 1 IDE Timing Register" on page 830	00h
48h	48h	"Offset 48h: SYNCC – Synchronous DMA Control Register" on page 831	00h
4Ah	4Bh	"Offset 4Ah: SYNCTIM – Synchronous DMA Timing Register" on page 832	0000h
54h	57h	"Offset 54h: IIOC – IDE I/O Configuration Register" on page 833	00000000h
70h	71h	"Offset 70h: PID – PCI Power Management Capability ID Register" on page 834	Variable
72h	73h	"Offset 72h: PC – PCI Power Management Capabilities Register" on page 834	4002h
74h	77h	"Offset 74h: PMCS – PCI Power Management Control And Status Register" on page 835	0000h
80h	81h	"Offset 80h: MID – Message Signaled Interrupt Identifiers Register" on page 836	7005h
82h	83h	"Offset 82h: MC – Message Signaled Interrupt Message Control Register" on page 837	0000h
84h	87h	"Offset 84h: MA – Message Signaled Interrupt Message Address Register" on page 838	00000000h
88h	89h	"Offset 88h: MD – Message Signaled Interrupt Message Data Register" on page 838	0000h
90h	90h	"Offset 90h: MAP – Port Mapping Register" on page 839	00h

Table 7-27. Bus 0, Device 31, Function 2: Summary of SATA Controller PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
92h	92h	"Offset 92h: PCS – Port Control and Status Register" on page 840	00h
A8h	ABh	"Offset A8h: SATACR0 – Serial ATA Capability Register 0" on page 841	00100012h
ACh	AFh	"Offset ACh: SATACR1 – Serial ATA Capability Register 1" on page 841	00000048h
C0h	C0h	"Offset C0h: ATC – APM Trapping Control Register" on page 842	00h
C4h	C4h	"Offset C4h: ATS – ATM Trapping Status Register" on page 843	00h
D0h	D3h	"Offset D0h: SP – Scratch Pad Register" on page 844	00000000h
E0h	E3h	"Offset E0h: BFCS – BIST FIS Control/Status Register" on page 844	00000000h
E4h	E7h	"Offset E4h: BFTD1 – BIST FIS Transmit Data 1 Register" on page 846	00000000h
E8h	EBh	"Offset E8h: BFTD2 – BIST FIS Transmit Data 2 Register" on page 846	0h
F8h	FBh	"Offset F8h: MANID – Manufacturing ID Register" on page 847	Variable

Table 7-28. Bus 0, Device 31, Function 2: Summary of SATA Controller Configuration Registers Mapped Through LBAR I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PCMD – Primary Command Register" on page 848	00h
02h	02h	"Offset 02h: PSTS – Primary Status Register" on page 849	00h
04h	07h	"Offset 04h: PDTP – Primary Descriptor Table Pointer Register" on page 849	Variable
10h	13h	"Offset 10h: INDEX – AHCI Index Register" on page 850	00000000h
14h	17h	"Offset 14h: DATA – AHCI Data Register" on page 851	Variable

Table 7-29. Bus 0, Device 31, Function 2: Summary of SATA Controller Configuration Registers Mapped Through ABAR Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: HCAP – HBA Capabilities Register" on page 853	Variable
04h	07h	"Offset 04h: GHC – Global HBA Control Register" on page 855	00000000h
08h	0Bh	"Offset 08h: IS – Interrupt Status Register" on page 856	00000000h
0Ch	0Fh	"Offset 0Ch: PI – Ports Implemented Register" on page 856	00000000h
10h	13h	"Offset 10h: VS – AHCI Version Register" on page 857	00010100h
A0h	A3h	"Offset A0h: SGPO -SPGIO Control Register" on page 857	00000000h
100h, 180h	17Fh, 1FFh	"Offset 100h: PxCLB[0-1] – Port [0-1] Command List Base Address Register" on page 858	Variable
104h, 184h	107h, 187h	"Offset 104h: PxCLBU[0-1] – Port [0-1] Command List Base Address Register" on page 858	Variable
108h, 188h	10Bh, 18Bh	"Offset 108h: PxFB[0-1] – Port [0-1] FIS Base Address Register" on page 859	Variable

Table 7-29. Bus 0, Device 31, Function 2: Summary of SATA Controller Configuration Registers Mapped Through ABAR Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
10Ch, 18Ch	10Fh, 18Fh	"Offset 10Ch: PxFBU[0-1] – Port [0-1] FIS Base Address Upper 32-bits Register" on page 859	Variable
110h, 190h	113h, 193h	"Offset 110h: PxIS[0-1] – Port [0-1] Interrupt Status Register" on page 860	00000000h
114h, 194h	117h, 197h	"Offset 114h: PxIE[0-1] – Port [0-1] Interrupt Enable Register" on page 861	00000000h
118h, 198h	11Bh, 19Bh	"Offset 118h: PxCMD[0-1] – Port [0-1] Command Register" on page 863	Variable
120h, 1A0h	123h, 1A3h	"Offset 120h: PxTFD[0-1] – Port [0-1] Task File Data Register" on page 866	0000007Fh
124h, 1A4h	127h, 1A7h	"Offset 124h: PxSIG[0-1] – Port [0-1] Signature Register" on page 867	FFFFFFFFh
128h, 1A8h	12Bh, 1ABh	"Offset 128h: PxSSTS[0-1] – Port [0-1] Serial ATA Status Register" on page 868	Variable
12Ch, 1ACh	12Fh, 1AFh	"Offset 12Ch: PxSCTL[0-1] – Port [0-1] Serial ATA Control Register" on page 869	00000000h
130h, 1B0h	133h, 1B3h	"Offset 130h: PxSERR[0-1] – Port [0-1] Serial ATA Error Register" on page 870	00000000h
134h, 1B4h	137h, 1B7h	"Offset 134h: PxSACT[0-1] – Port [0-1] Serial ATA Active Register" on page 872	00000000h
138h, 1B8h	13Bh, 1BBh	"Offset 138h: PxCI[0-1] – Port [0-1] Command Issue Register" on page 872	00000000h
13Ch, 1BCh	13Fh, 1BFh	"Offset 13Ch: PxSNTF[0-1] – Port [0-1] SNotification Register" on page 873	00000000h

7.3.11 SMBus Controller: Bus 0, Device 31, Function 3

The SMBus controller includes the registers listed in Table 7-30 and Table 7-31. These registers materialize in PCI configuration and I/O spaces (via PCI I/O BAR), respectively. See Chapter 24.0, "SMBus Controller Functional Description: Bus 0, Device 31, Function 3" for detailed discussion of these registers.

Table 7-30. Bus 0, Device 31, Function 3: Summary of SMBus Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor ID Register" on page 897	8086h
02h	03h	"Offset 02h: DID: Device ID Register" on page 897	5032h
04h	05h	"Offset 04h: CMD: Command Register" on page 897	0000h
06h	07h	"Offset 06h: DS – Device Status Register" on page 898	0280h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 899	Variable
09h	09h	"Offset 09h: PI: Programming Interface Register" on page 900	00h
0Ah	0Ah	"Offset 0Ah: SCC: Sub Class Code Register" on page 900	05h
0Bh	0Bh	"Offset 0Bh: BCC: Base Class Code Register" on page 900	0Ch
20h	23h	"Offset 20h: SM_BASE: SMB Base Address Register" on page 901	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: SVID Register" on page 901	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem Identification Register" on page 902	0000h
3Ch	3Ch	"Offset 3Ch: INTLN: Interrupt Line Register" on page 902	00h
3Dh	3Dh	"Offset 3Dh: NTPN: Interrupt Pin Register" on page 903	Variable
40h	40h	"Offset 40h: HCFG: Host Configuration Register" on page 903	00h
F8h	FBh	"Offset F8h: MANID: Manufacturer ID Register" on page 904	00010F90h

Table 7-31. Bus 0, Device 31, Function 3: Summary of SMBus Controller Configuration Registers Mapped Through SM_BASE I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: HSTS: Host Status Register" on page 906	00h
02h	02h	"Offset 02h: HCTL: Host Control Register" on page 908	00h
03h	03h	"Offset 03h: HCMD: Host Command Register" on page 912	00h
04h	04h	"Offset 04h: TSA: Transmit Slave Address Register" on page 912	00h
05h	05h	"Offset 05h: HD0: Data 0 Register" on page 913	00h
06h	06h	"Offset 06h: HD1: Data 1 Register" on page 913	00h
07h	07h	"Offset 07h: HBD: Host Block Data Register" on page 914	00h
08h	08h	"Offset 08h: PEC: Packet Error Check Data Register" on page 915	00h
0Ch	0Ch	"Offset 0Ch: AUXS: Auxiliary Status Register" on page 915	00h
0Dh	0Dh	"Offset 0Dh: AUXC: Auxiliary Control Register" on page 916	00h
0Eh	0Eh	"Offset 0Eh: SMLC: SMLINK_PIN_CTL Register" on page 916	07h
0Fh	0Fh	"Offset 0Fh: SMBC: SMBUS_PIN_CTL Register" on page 917	07h

7.3.12 IA-32 Core Interface I/O-Mapped Register

Table 7-32. Summary of IA-32 Core Interface Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
61h	61h	"Offset 61h: NMI_STS_CNT - NMI Status and Control Register" on page 1098	00h
70h	70h	"Offset 70h: NMI_EN - NMI Enable (and Real Time Clock Index) Register" on page 1099	80h
92h	92h	"Offset 92h: PORT92 - Fast A20 and Init Register" on page 1100	00h
F0h	F0h	"Offset F0h: COPROC_ERR - Coprocessor Error Register" on page 1100	00h
CF9h	CF9h	"Offset CF9h: RST_CNT - Reset Control Register" on page 1101	00h

7.3.13 IMCH PCI Configuration

The PCI configuration interface includes the registers listed in Table 7-33. These registers materialize at fixed locations in I/O space.

Table 7-33. Summary of IMCH PCI Configuration Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
0CF8h	0CF8h	"Offset 0CF8h: CONFIG_ADDRESS: Configuration Address Register" on page 354	00000000h
0CFCh	0CFCh	"Offset 0CFCh: CONFIG_DATA: Configuration Data Register" on page 355	00000000h

7.3.14 APIC

The APIC includes the registers listed in Table 7-34 and Table 7-35 These registers materialize at fixed locations in memory space and are indexed, respectively. See Chapter 30.0 for detailed discussion of these registers.

Table 7-34. Summary of APIC Registers Mapped in Memory Space"

Offset Start	Offset End	Register ID - Description	Default Value
0000h (4B)	0000h (4B)	"APIC_IDX - Index Register" on page 1135	00h
0010h (4B)	0010h (4B)	"APIC_DAT – Data Register" on page 1136	00h
0040h (4B)	0040h (4B)	"APIC_EOI - EOI Register" on page 1136	00h

Table 7-35. Summary of APIC Indexed Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h (4B)	00h (4B)	“APIC_ID – Identification Register” on page 1138	0000h
01h (4B)	01h (4B)	“APIC_VS - Version Register” on page 1138	00170020h
10h at 02h (4B)	11h at 02h (4B)	“APIC_RTE[0-39] - Redirection Table Entry” on page 1139	XXXX0000000 1XXXXh

7.3.15 8259 Interrupt Controller (PIC)

The 8259 Interrupt Controller includes the registers listed in Table 7-36. These registers materialize at fixed locations in I/O space. See Chapter 30.0, "Interrupts" for detailed discussion of these registers.

Table 7-36. Summary of 8259 Interrupt Controller (PIC) Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
020h, 0A0h	020h, 0A0h	"ICW1[0-1] - Initialization Command Word 1 Register" on page 1119	0001X0XXb
021h, 0A1h	021h, 0A1h	"ICW2[0-1] - Initialization Command Word 2 Register" on page 1120	XXh
21h	21h	"MICW3 - Master Initialization Command Word 3 Register" on page 1121	04h
A1h	A1h	"SICW3 - Slave Initialization Command Word 3 Register" on page 1121	00h
21h, 0A1h	21h, 0A1h	"ICW4[0-1] - Initialization Command Word 4 Register" on page 1122	01h
021h, 0A1h	021h, 0A1h	"OCW1[0-1]- Operational Control Word 1 (Interrupt Mask) Register" on page 1122	00h
020h, 0A0h	020h, 0A0h	"OCW2[0-1] - Operational Control Word 2 Register" on page 1123	001XXXXXb
020h, 0A0h	020h, 0A0h	"OCW3[0-1] - Operational Control Word 3 Register" on page 1124	001XX10b
4D0h	4D0h	"ELCR1 - Master Edge/Level Control Register" on page 1125	00h
4D1h	4D1h	"ELCR2 - Slave Edge/Level Control Register" on page 1126	00

7.3.16 APM Power Management

The APM power management includes the registers listed in Table 7-37. These registers materialize at fixed locations in I/O space. See Chapter 27.0, "Power Management" for detailed discussion of these registers.

Table 7-37. Summary of APM Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
B2h	B2h	"Offset B2h: APM_CNT - Advanced Power Management Control Port Register" on page 1054	00h
B3h	B3h	"Offset B3h: APM_STS - Advanced Power Management Status Port Register" on page 1054	00h

7.3.17 LPC DMA

The LPC DMA interface includes the registers listed in Table 7-38 through Table 7-40. These registers materialize at fixed locations in I/O space. See Chapter 20.0, "LPC DMA" for detailed discussion of these registers.

Table 7-38. Summary of LPC DMA Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
00h at 02h	10h at 02h	"Offset 00h: DMA_BCA[0-3] - DMA Base and Current Address Registers for Channels 0-3" on page 766	XXXX
C4h at 04h	C5h at 04h	"Offset C4h: DMA_BCA[5-7] - DMA Base and Current Address Registers for Channels 5-7" on page 767	XXXX
01h at 02h	11h at 02h	"Offset 01h: DMA_BCC[0-3] - DMA Base and Current Count Registers for Channels 0-3" on page 768	XXXX
C6h at 04h	C7h at 04h	"Offset C6h: DMA_BCC[5-7] - DMA Base and Current Count Registers for Channels 5-7" on page 769	XXXX
87h, 83h, 81h, 82h	97h, 93h, 91h, 82h	"Offset 87h: DMA_MPL[0-3] - DMA Memory Low Page Registers for Channels 0-3" on page 771	XXXXXXX
8Bh, 89h, 8Ah	9Bh, 99h, 9Ah	"Offset 8Bh: DMA_MPL[5-7]: DMA Memory Low Page Registers for Channels 5-7" on page 771	XXXXXXX

Table 7-39. 0000h (IO) Base Address Registers in the IA F1 View

Offset Start	Offset End	Register ID - Description	Default Value
08h	08h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
18h	18h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
08h	08h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
18h	18h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
0Ah	0Ah	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
1Ah	1Ah	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
0Bh	0Bh	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
1Bh	1Bh	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
0Ch	0Ch	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
1Ch	1Ch	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
0Dh	0Dh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
1Dh	1Dh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
0Eh	0Eh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
1Eh	1Eh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
0Fh	0Fh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b
1Fh	1Fh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b

Table 7-40. 0000h (IO) Base Address Registers in the IA F2 View

Offset Start	Offset End	Register ID - Description	Default Value
D0h	D0h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
D1h	D1h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
D0h	D0h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
D1h	D1h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
D4h	D4h	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
D5h	D5h	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
D6h	D6h	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
D7h	D7h	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
D8h	D8h	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
D9h	D9h	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
DAh	DAh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
DBh	DBh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
DCh	DCh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
DDh	DDh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
DEh	DEh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b
DFh	DFh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b

7.3.18 8254 Timers

The 8254 timers include the registers listed in Table 7-41. These registers materialize at fixed locations in I/O space. See Chapter 31.0, "8254 Timers" for detailed discussion of these registers.

Table 7-41. Summary of 8254 Timer Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
43h	43h	"Offset 43h: TCW - Timer Control Word Register" on page 1146	XXh
40h at 01h	40h at 01h	"Offset 40h: TSB[0-2] - Interval Timer Status Byte Format Register" on page 1147	0XXXXXXXb
40h at 01h	40h at 01h	"Offset 40h: TCAP[0-2] - Interval Timer Counter Access Ports Register" on page 1148	XXh

7.3.19 High Precision Event Timers

The High Precision Event Timers includes the registers listed in Table 7-42. These registers materialize at fixed locations in memory space. See Chapter 32.0, "High Precision Event Timers" for detailed discussion of these registers.

Table 7-42. Summary of HPET Registers Mapped in Memory Space

Offset Start	Offset End	Register ID - Description	Default Value
000h	007h	"Offset 000h: GCAP_ID - General Capabilities and ID Register" on page 1155	0429B17F8086A201h
010h	017h	"Offset 010h: GEN_CONF - General Configuration Register" on page 1156	0000000000000000h
020h	027h	"Offset 020h: GINTR_STA - General Interrupt Status Register" on page 1157	0000000000000000h
0F0h	0F7h	"Offset 0F0h: MAIN_CNT - Main Counter Value Register" on page 1158	Xh
100h at 20h	107h at 20h	"Offset 100h: HPTCC[0-2] - Timer n Configuration and Capabilities Register" on page 1159	Xh
108h at 20h	10Fh at 20h	"Offset 108h: HPTCV[0-2] - Timer n Comparator Value Register" on page 1163	Xh

7.3.20 Watchdog Timer and Serial I/O

The Watchdog Timers and Serial I/O units includes the registers listed in Table 7-43 and Table 7-44. These registers materialize at fixed locations in I/O space. See Chapter 33.0, "Serial I/O Unit and Watchdog Timer" for detailed discussion of these registers.

Table 7-43. Summary of UART Timer registers in I/O space

Offset Start	Offset End	Register ID - Description	Default Value
02h	02h	"Offset 02h: IIR - Interrupt Identification Register" on page 1179	01h
02h	02h	"Offset 02h: FCR - FIFO Control Register" on page 1180	00h
03h	03h	"Offset 03h: LCR - Line Control Register" on page 1182	00h
04h	04h	"Offset 04h: MCR - Modem Control Register" on page 1184	00h
05h	05h	"Offset 05h: LSR - Line Status Register" on page 1186	60h
06h	06h	"Offset 06h: MSR - Modem Status Register" on page 1189	00h
07h	07h	"Offset 07h: SCR - Scratchpad Register" on page 1190	00h

Table 7-44. Summary of Watchdog Timer Registers in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PV1R0 - Preload Value 1 Register 0" on page 1194	FFh
01h	01h	"Offset 01h: PV1R1 - Preload Value 1 Register 1" on page 1195	FFh
02h	02h	"Offset 02h: PV1R2 - Preload Value 1 Register 2" on page 1195	0Fh
04h	04h	"Offset 04h: PV2R0 - Preload Value 2 Register 0" on page 1196	FFh
05h	05h	"Offset 05h: PV2R1 - Preload Value 2 Register 1" on page 1196	FFh
06h	06h	"Offset 06h: PV2R2 - Preload Value 2 Register 2" on page 1197	0Fh
08h	08h	"Offset 08h: GISR - General Interrupt Status Register" on page 1197	00h
0Ch	0Ch	"Offset 0Ch: RR0 - Reload Register 0" on page 1198	00h
0Dh	0Dh	"Offset 0Dh: RR1 - Reload Register 1" on page 1199	00h
10h	10h	"Offset 10h: WDTCR - WDT Configuration Register" on page 1199	00h
18h	18h	"Offset 18h: WDTLR - WDT Lock Register" on page 1201	00h

7.3.21 Real Time Clock

The Real Time Clock include the registers listed in Table 7-45. These registers materialize at indexed locations. See Chapter 29.0, "Real Time Clock" for detailed discussion of these registers.

Table 7-45. Summary of Real Time Clock Indexed Registers

Offset Start	Offset End	Register ID - Description	Default Value
0Ah	0Ah	"Offset 0Ah: RTC_REGA - Register A (General Configuration)" on page 1107	XXh
0Bh	0Bh	"Offset 0Bh: RTC_REGB - Register B (General Configuration)" on page 1109	X0X00XXXb
0Ch	0Ch	"Offset 0Ch: RTC_REGC - Register C (Flag Register)" on page 1110	00X00000b
0Dh	0Dh	"Offset 0Dh: RTC_REGD - Register D (Flag Register)" on page 1111	10XXXXXXb

7.4 AIOC Registers

This section summarizes the registers in the AIOC. The registers are presented as they materialize from a PCI perspective.

7.4.1 PCI-to-PCI Bridge: Bus 0, Device 4, Function 0

The PCI-to-PCI Bridge includes the registers listed in Table 7-46. These registers materialize in PCI configuration space. See Chapter 34.0, "PCI-to-PCI Bridge Detailed Register Descriptions" for detailed discussion of these registers along with alternative materializations.

Table 7-46. Bus 0, Device 4, Function 0: Summary of PCI-to-PCI Bridge PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
0h	1h	"Offset 0h: VID: Vendor Identification Register" on page 1217	8086h
2h	3h	"Offset 2h: DID: Device Identification Register" on page 1217	5037h
4h	5h	"Offset 4h: PCICMD: Device Command Register" on page 1217	0h
6h	7h	"Offset 6h: PCISTS: PCI Device Status Register" on page 1218	10h
8h	8h	"Offset 8h: RID: Revision ID Register" on page 1219	Variable
9h	Bh	"Offset 9h: CC: Class Code Register" on page 1219	060400h
Ch	Ch	"Offset Ch: CLS: Cacheline Size Register" on page 1219	00h
Dh	Dh	"Offset Dh: LT: Latency Timer Register" on page 1220	00h
Eh	Eh	"Offset Eh: HDR: Header Type Register" on page 1220	1h
10h	14h	"Offset 10h: CSRBAR0: Control and Status Registers Base Address Register" on page 1220	00h
14h	17h	"Offset 14h: CSRBAR1: Control and Status Registers Base Address Register" on page 1221	00h
18h	18h	"Offset 18h: PBNUM: Primary Bus Number Register" on page 1221	00h
19h	19h	"Offset 19h: SECBNM: Secondary Bus Number Register" on page 1221	00h
1Ah	1Ah	"Offset 1Ah: SUBBNM: Subordinate Bus Number Register" on page 1222	00h
1Bh	1Bh	"Offset 1Bh: SECLT: Secondary Latency Timer Register" on page 1222	00h
1Ch	1Ch	"Offset 1Ch: IOB: I/O Base Register" on page 1222	F0
1Dh	1Dh	"Offset 1Dh: IOL: I/O Limit Register" on page 1223	0
1Eh	1Fh	"Offset 1Eh: SECSTA: Secondary Status Register" on page 1223	0h
20h	21h	"Offset 20h: MEMB: Memory Base Register" on page 1224	FFF0
22h	23h	"Offset 22h: MEML: Memory Limit Register" on page 1224	0
24h	25h	"Offset 24h: PMASE: Prefetchable Memory Base Register" on page 1225	FFF1H
26h	27h	"Offset 26h: PMLIMIT: Prefetchable Memory Limit Register" on page 1225	1H
28h	28h	"Offset 28h: PMBASU: Memory Limit Register" on page 1226	Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU: Prefetchable Memory Limit Upper Register" on page 1226	0
30h	31h	"Offset 30h: IOBU: I/O Base Upper Register" on page 1227	0
32h	33h	"Offset 32h: IOLU: I/O Limit Upper Register" on page 1227	0
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1227	dch
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1228	0

Table 7-46. Bus 0, Device 4, Function 0: Summary of PCI-to-PCI Bridge PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1228	0
3Eh	3Fh	"Offset 3Eh: BCTL: Bridge Control Register" on page 1228	0000h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1229	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1230	00h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1230	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1231	0008h
E2h	E2h	"Offset E2h: PMCSE: Power Management Control and Status Extension Register" on page 1232	0000h

7.4.2 Gigabit Ethernet MAC: Bus M, Devices 0, 1, and 2, Function 0

The Gigabit Ethernet MAC includes the registers listed in Table 7-47 through Table 7-44. These registers materialize in PCI configuration, I/O (via PCI BAR), and memory (via PCI BAR) spaces. See Section 35.6, "Gigabit Ethernet MAC Configuration Spaces: Bus M, Device 0-2, Function 0", Section 35.7, "Gigabit Ethernet MAC I/O Spaces: Bus M, Device 0-2, Function 0", and Section 37.6, "GbE Controller Register Summary" for detailed discussion of these registers along with alternative materializations.

Table 7-47. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1241	5040h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h

Table 7-47. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

Table 7-48. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1242	5044h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h

Table 7-48. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

Table 7-49. Bus M, Device2, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1242	5048h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h

Table 7-49. Bus M, Device2, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

Table 7-50. Bus M, Device 0, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 7-51. Bus M, Device 1, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 7-52. Bus M, Device 2, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 7-53. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"CTRL: Device Control Register" on page 1438	00000A09h
0008h	000Bh	"STATUS: Device Status Register" on page 1441	0000XXXXh
0018h	001Bh	"CTRL_EXT: Extended Device Control Register" on page 1442	00000000h
00E0h	00E3h	"CTRL_AUX: Auxiliary Device Control Register" on page 1444	00000100h
0010h	0013h	"EEPROM_CTRL - EEPROM Control Register" on page 1446	00000X1Xh
0014h	0017h	"EEPROM_RR – EEPROM Read Register" on page 1448	XXXXXX00h
0028h	002Bh	"FCAL: Flow Control Address Low Register" on page 1449	00c28001h
002Ch	002Fh	"FCAH: Flow Control Address High Register" on page 1450	00000100h

Table 7-53. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0030h	0033h	"FCT: Flow Control Type Register" on page 1451	00008808h
0038h	003Bh	"VET: VLAN EtherType Register" on page 1452	00008100h
0170h	0173h	"FCTTV: Flow Control Transmit Timer Value Register" on page 1452	00000000h
1000h	1003h	"PBA: Packet Buffer Allocation Register" on page 1453	00100030h
00C0h	00C3h	"ICR0: Interrupt 0 Cause Read Register" on page 1454	00000000h
00C4h	00C7h	"ITR0: Interrupt 0 Throttling Register" on page 1457	00000000h
00C8h	00CBh	"ICS0: Interrupt 0 Cause Set Register" on page 1458	00000000h
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh

Table 7-53. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h

Table 7-53. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5843h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh
05884h	5887h	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

Table 7-54. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"CTRL: Device Control Register" on page 1438	00000A09h
0008h	000Bh	"STATUS: Device Status Register" on page 1441	0000XXXXh
0018h	001Bh	"CTRL_EXT: Extended Device Control Register" on page 1442	00000000h
00E0h	00E3h	"CTRL_AUX: Auxiliary Device Control Register" on page 1444	00000100h
0010h	0013h	"EEPROM_CTRL - EEPROM Control Register" on page 1446	00000X1Xh
0014h	0017h	"EEPROM_RR – EEPROM Read Register" on page 1448	XXXXXX00h
0028h	002Bh	"FCAL: Flow Control Address Low Register" on page 1449	00c28001h
002Ch	002Fh	"FCAH: Flow Control Address High Register" on page 1450	00000100h
0030h	0033h	"FCT: Flow Control Type Register" on page 1451	00008808h
0038h	003Bh	"VET: VLAN EtherType Register" on page 1452	00008100h
0170h	0173h	"FCTTV: Flow Control Transmit Timer Value Register" on page 1452	00000000h
1000h	1003h	"PBA: Packet Buffer Allocation Register" on page 1453	00100030h
00C0h	00C3h	"ICR0: Interrupt 0 Cause Read Register" on page 1454	00000000h
00C4h	00C7h	"ITR0: Interrupt 0 Throttling Register" on page 1457	00000000h
00C8h	00CBh	"ICS0: Interrupt 0 Cause Set Register" on page 1458	00000000h
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h

Table 7-54. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h

Table 7-54. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5843h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh

Table 7-54. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
05884h	5887h	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

Table 7-55. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"CTRL: Device Control Register" on page 1438	00000A09h
0008h	000Bh	"STATUS: Device Status Register" on page 1441	0000XXXXh
0018h	001Bh	"CTRL_EXT: Extended Device Control Register" on page 1442	00000000h
00E0h	00E3h	"CTRL_AUX: Auxiliary Device Control Register" on page 1444	00000100h
0010h	0013h	"EEPROM_CTRL - EEPROM Control Register" on page 1446	00000X1Xh
0014h	0017h	"EEPROM_RR – EEPROM Read Register" on page 1448	XXXXXX00h
0028h	002Bh	"FCAL: Flow Control Address Low Register" on page 1449	00c28001h
002Ch	002Fh	"FCAH: Flow Control Address High Register" on page 1450	00000100h
0030h	0033h	"FCT: Flow Control Type Register" on page 1451	00008808h
0038h	003Bh	"VET: VLAN EtherType Register" on page 1452	00008100h
0170h	0173h	"FCTTV: Flow Control Transmit Timer Value Register" on page 1452	00000000h
1000h	1003h	"PBA: Packet Buffer Allocation Register" on page 1453	00100030h
00C0h	00C3h	"ICR0: Interrupt 0 Cause Read Register" on page 1454	00000000h
00C4h	00C7h	"ITR0: Interrupt 0 Throttling Register" on page 1457	00000000h
00C8h	00CBh	"ICS0: Interrupt 0 Cause Set Register" on page 1458	00000000h
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h

Table 7-55. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h

Table 7-55. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h

Table 7-55. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5607h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh
05884h	0588Fh	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

7.4.3 GCU: Bus M, Device 3, Function 0

The GCU includes the registers listed in Table 7-56 and Table 7-57. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 35.8, "GCU Configuration Space: Bus M, Device 3, Function 0"and Chapter 38.0, "Register Summary" for detailed discussion of these registers along with alternative materializations.

Table 7-56. Bus M, Device 3, Function 0: Summary of GCU PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1265	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1266	503Eh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1266	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1267	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1268	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1268	FF0000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1268	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1269	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1269	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1270	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1270	DCh
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1270	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1271	00h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1271	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1272	0000h

Table 7-57. Bus M, Device 3, Function 0: Summary of GCU Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000010h	00000013h	"Offset 0x00000010h: MDIO_STATUS - MDIO Status Register" on page 1562	00000000h
00000014h	00000017h	"Offset 0x00000014h: MDIO_COMMAND - MDIO Command Register" on page 1562	00000000h
00000018h	0000001Bh	"Offset 0x00000018h: MDIO_DRIVE - MDIO Drive Register" on page 1563	03030107h
00000020h	00000023h	"Offset 0x00000020h: MDC_DRIVE - MDC Drive Register" on page 1563	0303030Fh
00000024h	00000027h	"Offset 0x00000024h: GCU_GBE_RC_CTRL - GCU GbE RCOMP Control Register" on page 1564	0031F31Fh
00000044h	00000047h	"Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register" on page 1564	00000000h
00000050h	00000053h	"Offset 0x00000050h: GCU_LEB_RC_STAT - GCU Local Expansion Bus RCOMP Status Register" on page 1565	63000300h

Table 7-57. Bus M, Device 3, Function 0: Summary of GCU Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000054h	00000057h	"Offset 0x00000054h: GCU_LEB_RC_CTRL - GCU Local Expansion Bus RCOMP Control Register" on page 1566	000030F301h
00000060h	00000063h	"Offset 0x00000060h: SSP_DRIVE - SSP Drive Register" on page 1566	02000200h
00000064h	00000067h	"Offset 0x00000064h: TDM_DRIVE_3 - TDM Drive Register for TDM ports 3" on page 1567	02000200h
00000068h	0000006Bh	"Offset 0x00000068h: TDM_DRIVE_12 - TDM Drive Register for TDM ports 1 & 2" on page 1567	02000200h
00000028h	0000002Bh	"Offset 0x00000028h: CAN_DRIVE - CAN Drive Register" on page 1568	02000200h

7.4.4 CAN Interface: Bus M, Device 4 and 5, Function 0

The CAN interface includes the registers listed in Table 7-58 through Table 7-61. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 35.9, "CAN Controller Configuration Spaces: Bus M, Device 4-5, Function 0" and Chapter 39.0, "Detailed Register Descriptions" for detailed discussion of these registers along with alternative materializations.

Table 7-58. Bus M, Device 4, Function 0: Summary of CAN Interface PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1275	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1275	5039h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1276	0h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1277	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1278	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1278	0C0900h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1279	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1279	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1280	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1280	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1281	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1281	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1282	01h
40h	40h	"Offset 40h: CANCTL - CAN Control Register" on page 1282	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1283	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1283	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1284	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1284	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1285	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1285	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1286	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1286	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1287	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1287	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1288	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1288	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1289	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1289	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1290	0000h

Table 7-59. Bus M, Devices 5, Function 0: Summary of CAN Interface PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1275	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1276	503Ah
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1276	0h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1277	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1278	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1278	0C0900h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1279	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1279	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1280	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1280	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1281	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1281	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1282	01h
40h	40h	"Offset 40h: CANCTL - CAN Control Register" on page 1282	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1283	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1283	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1284	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1284	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1285	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1285	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1286	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1286	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1287	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1287	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1288	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1288	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1289	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1289	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1290	0000h

Table 7-60. Bus M, Device 4, Function 0: Summary of CAN Registers Mapped Through CSRBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 00000000h: Int_Status - Interrupt Status Register" on page 1587	00000000h
00000004h	00000007h	"Offset 00000004h: Int_Ebl - Interrupt Enable Register" on page 1588	00000000h
00000008h	0000000Ah	"Offset 00000008h: Buffer Status Indicators" on page 1589	00000000h
0000000Ch	0000000Fh	"Offset 0000000Ch: ErrorStatus - Error Status Indicators" on page 1590	00000000h
00000010h	00000013h	"Offset 00000010h: Command - Operating Modes" on page 1591	00000000h
00000014h	00000017h	"Offset 00000014h: Config - CAN Configuration Register" on page 1592	00000000h
00000020h at 10h	00000023h at 10h	"Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command" on page 1593	XXXXXXXXh
00000024h at 10h	00000027h at 10h	"Offset 00000024h: TxMessageID[0-7] - Transmit Message ID" on page 1595	XXXXXXXXh
00000028h at 10h	0000002Ah at 10h	"Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High" on page 1596	XXXXXXXXh
0000002Ch at 10h	0000002Fh at 10h	"Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low" on page 1597	XXXXXXXXh
000000A0h at 20h	000000A3h at 20h	"Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control" on page 1598	XXXXXXXXh
000000A4h at 20h	000000A7h at 20h	"Offset 000000A4h: RxMessageID[0-15] - Receive Message ID" on page 1600	XXXXXXXXh
000000A8h at 20h	000000ABh at 20h	"Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High" on page 1600	XXXXXXXXh
000000ACh at 20h	000000AFh at 20h	"Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low" on page 1601	XXXXXXXXh
000000B0h at 20h	000000B3h at 20h	"Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR" on page 1601	XXXXXXXh
000000B4h at 20h	000000B7h at 20h	"Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR" on page 1602	XXXXXXXXh
000000B8h at 20h	000000BBh at 20h	"Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data" on page 1603	XXXXXXXXh
000000BCh at 20h	000000BFh at 20h	"Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data" on page 1604	XXXXXXXXh

Table 7-61. Bus M, Device 5, Function 0: Summary of CAN Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 00000000h: Int_Status - Interrupt Status Register" on page 1587	00000000h
00000004h	00000007h	"Offset 00000004h: Int_Ebl - Interrupt Enable Register" on page 1588	00000000h
00000008h	0000000Ah	"Offset 00000008h: Buffer Status Indicators" on page 1589	00000000h
0000000Ch	0000000Fh	"Offset 0000000Ch: ErrorStatus - Error Status Indicators" on page 1590	00000000h
00000010h	00000013h	"Offset 00000010h: Command - Operating Modes" on page 1591	00000000h
00000014h	00000017h	"Offset 00000014h: Config - CAN Configuration Register" on page 1592	00000000h

Table 7-61. Bus M, Device 5, Function 0: Summary of CAN Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000020h at 10h	00000023h at 10h	"Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command" on page 1593	XXXXXXXXh
00000024h at 10h	00000027h at 10h	"Offset 00000024h: TxMessageID[0-7] - Transmit Message ID" on page 1595	XXXXXXXXh
00000028h at 10h	0000002Ah at 10h	"Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High" on page 1596	XXXXXXXXh
0000002Ch at 10h	0000002Fh at 10h	"Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low" on page 1597	XXXXXXXXh
000000A0h at 20h	000000A3h at 20h	"Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control" on page 1598	XXXXXXXXh
000000A4h at 20h	000000A7h at 20h	"Offset 000000A4h: RxMessageID[0-15] - Receive Message ID" on page 1600	XXXXXXXXh
000000A8h at 20h	000000ABh at 20h	"Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High" on page 1600	XXXXXXXXh
000000ACh at 20h	000000AFh at 20h	"Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low" on page 1601	XXXXXXXXh
000000B0h at 20h	000000B3h at 20h	"Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR" on page 1601	XXXXXXXh
000000B4h at 20h	000000B7h at 20h	"Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR" on page 1602	XXXXXXXXh
000000B8h at 20h	000000BBh at 20h	"Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data" on page 1603	XXXXXXXXh
000000BCh at 20h	000000BFh at 20h	"Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data" on page 1604	XXXXXXXXh

7.4.5 SSP Interface: Bus M, Device 6, Function 0

The SSP interface includes the registers listed in Table 7-62 and Table 7-63. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 35.10, "SSP Controller Configuration Space: Bus M, Device 6, Function 0" and Table 40.4, "Register Summary" on page 1606 for detailed discussion of these registers along with alternative materializations.

Table 7-62. Bus M, Device 6, Function 0: Summary of SSP Controller PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1292	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1292	503Bh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1292	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1293	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1294	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1295	078000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1295	00h

Table 7-62. Bus M, Device 6, Function 0: Summary of SSP Controller PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1295	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1296	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1296	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1297	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1297	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1297	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1298	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1298	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1298	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1299	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1300	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1300	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1300	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1301	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1301	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1302	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1302	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1302	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1303	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1303	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1304	0000h

Table 7-63. Bus M, Device 6, Function 0: Summary of SSP CSRs

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: SSCR0 - SSP Control Register 0 Details" on page 1607	00000000h
04h	07h	"Offset 04h: SSCR1 - SSP Control Register 1 Details" on page 1610	00000000h
08h	0Bh	"Offset 08h: SSSR - SSP Status Register Details" on page 1614	0000F004h
0Ch	0Fh	"Offset 0Ch: SSITR - SSP Interrupt Test Register Details" on page 1617	00000000
10h	13h	"Offset 10h: SSDR - SSP Data Register Details" on page 1618	00000000h

7.4.6 IEEE 1588 Timestamp Unit: Bus M, Device 7, Function 0

The IEEE 1588 Timestamp Unit includes the registers listed in Table 7-64 and Table 7-65. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 35.11, "IEEE 1588 Hardware Assist Unit Configuration Space: Bus M, Device 7, Function 0" and Chapter 41.0, "Register Summary" for detailed discussion of these registers along with alternative materializations.

Table 7-64. Bus M, Device 7, Function 0: Summary of IEEE 1588 Timestamp Unit PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1306	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1306	503Ch
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1306	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1307	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1308	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1308	111000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1309	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1309	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1310	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1310	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1310	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1311	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1311	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1312	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1312	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1313	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1313	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1314	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1314	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1314	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1315	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1315	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1316	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1316	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1317	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1317	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1318	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1318	0000h

Table 7-65. Bus M, Device 7, Function 0: Summary of IEEE 1588 TSYNC CSRs (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 0000h: TS_Control Register" on page 1639	00000000h
00000004h	00000007h	"Offset 0004h: TS_Event Register" on page 1641	0022h
00000008h	0000000Bh	"Offset 0008h: TS_Addend Register" on page 1643	0000h
0000000Ch	0000000Fh	"Offset 000Ch: TS_Accum Register" on page 1643	0000h
00000010h	00000013h	"Offset 0010h: TS_Test Register" on page 1644	0000h
00000014h	00000017h	"Offset 0014h: TS_PPS_Compare Register" on page 1646	FFFFFFFFh
00000018h	0000001Bh	"Offset 0018h: TS_RSysTimeLo Register" on page 1647	0000h
0000001Ch	0000001Fh	"Offset 001Ch: TS_RSysTimeHI Register" on page 1648	0000h
00000020h	00000023h	"Offset 0020h: TS_SysTimeLo Register" on page 1649	0000h
00000024h	00000027h	"Offset 0024h: TS_SysTimeHi Register" on page 1650	0000h
00000028h	0000002Bh	"Offset 0028h: TS_TrgtLo Register" on page 1650	0000h
0000002Ch	0000002Fh	"Offset 002Ch: TS_TrgtHi Register" on page 1651	0000h
00000030h	00000033h	"Offset 0030h: TS_ASMSLo Register" on page 1652	0000h
00000034h	00000037h	"Offset 0034h: TS_ASMSHi Register" on page 1653	0000h
00000038h	0000003Bh	"Offset 0038h: TS_AMMSLo Register" on page 1654	0000h
0000003Ch	0000003Fh	"Offset 003Ch: TS_AMMSHi Register" on page 1655	0000h
0040h at 20h	0043h at 20h	"Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)" on page 1656	0000h
0044h at 20h	0047h at 20h	"Offset 0044h: TS_CH_EVENT[0-7] - Time Synchronization Channel Event Register Per Ethernet Channel)" on page 1658	0000h
0048h at 20h	004Bh at 20h	"Offset 0048h: TS_TxSnapLo[0-7] - Transmit Snapshot Low Register (Per Ethernet Channel)" on page 1659	0000h
004Ch at 20h	004Fh at 20h	"Offset 004Ch: TS_TxSnapHi[0-7] - Transmit Snapshot High Register (Per Ethernet Channel)" on page 1660	0000h
0050h at 20h	0053h at 20h	"Offset 0050h: TS_RxSnapLo[0-7] - Receive Snapshot Low Register (Per Ethernet Channel)" on page 1661	0000h
0054h at 20h	0057h at 20h	"Offset 0054h: TS_RxSnapHi[0-7] - Receive Snapshot High Register (Per Ethernet Channel)" on page 1662	0000h
0058h at 20h	005Bh at 20h	"Offset 0058h: TS_SrcUUIDLo[0-7] - Source UUID0 Low Register (Per Ethernet Channel)" on page 1663	0000h
005Ch at 20h	005Fh at 20h	"Offset 005Ch: TS_SrcUUIDHI[0-7] - SequenceID/SourceUUID High Register (Per Ethernet Channel)" on page 1664	0000h
0140h at 10h	0143h at 10h	"Offset 0140h: TS_CANx_Status[0-1] - Time Synchronization Channel Event Register (Per CAN Channel)" on page 1665	0000h
0144h at 10h	0147h at 10h	"Offset 0144h: TS_CANSnapLo[0-1] - Transmit Snapshot Low Register (Per CAN Channel)" on page 1666	0000h
0148h at 10h	014Bh at 10h	"Offset 0148h: TS_CANSnapHi[0-1] - Transmit Snapshot High Register (Per CAN Channel)" on page 1667	0000h
000001F0h	000001F3h	"Offset 01F0h: TS_Aux_TrgtLo Register" on page 1668	0000h
000001F4h	000001F7h	"Offset 01F4h: TS_Aux_TrgtHi Register" on page 1668	0000h
00000200h	00000203h	"Offset 0200h: L2 EtherType Register" on page 1669	000088F7h

Table 7-65. Bus M, Device 7, Function 0: Summary of IEEE 1588 TSYNC CSRs (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
0000204h	0000207h	"Offset 0204h: User Defined EtherType Register" on page 1669	00000000h
00000208h	0000020Bh	"Offset 0208h:User Defined Header Offset Register" on page 1670	00000000h
0000020Ch	0000020Fh	"Offset 020Ch:User Defined Header Register" on page 1670	00000000h

7.4.7 Local Expansion Bus Interface: Bus M, Device 8, Function 0:

The Local Expansion Bus interface includes the registers listed in Table 7-66 and Table 7-67. These registers materialize in PCI configuration and memory (via PCI BAR) spaces. See Section 35.12, "Expansion Bus Configuration Space: Bus M, Device 8, Function 0", and Table 42.5, "Register Summary" on page 1696 for detailed discussion of these registers along with alternative materializations.

Table 7-66. Bus M, Device 8, Function 0: Summary of Local Expansion Bus PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1320	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1320	503Dh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1321	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1321	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1322	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1323	068000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1323	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1323	00000000h
14h	17h	"Offset 14h: MMBAR: Expansion Bus Base Address Register" on page 1324	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1325	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1325	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1326	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1326	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1326	01h
40h	43h	"Offset 40h: LEBCTL: LEB Control Register" on page 1327	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1327	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1328	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1328	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1329	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1329	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1330	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1330	09h

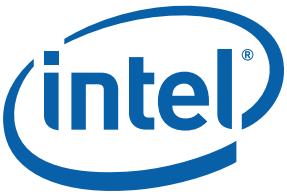
Table 7-66. Bus M, Device 8, Function 0: Summary of Local Expansion Bus PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1330	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1331	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1331	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1332	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1332	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1333	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1333	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1334	0000h

Table 7-67. Bus M, Device 8, Function 0: Summary of Local Expansion Bus Registers Mapped Through CSRBAR PCI Memory BAR"

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"EXP_TIMING_CS0 - Expansion Bus Timing Register" on page 1698	BFFF3C40h
00000004h at 4h	00000007h at 4h	"EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers" on page 1700	00000000h
00000020h	00000020h	"EXP_CNFG0 -Configuration Register 0" on page 1702	00000040h
00000120h	00000123h	"EXP_PARITY_STATUS - Expansion Bus Parity Status Register" on page 1703	00000000h

§ §



IA-32 Core and Integrated Memory Controller Hub, Volume 2 of 6

Order Number: 320066-003US

8.0 IA-32 Core

8.1 Overview

The EP80579system-on-a-chip (SoC) uses an IA-32 core, which is based on the Intel® Pentium® M processor (90 nm). Major features include:

- 600 MHz, 1066 MHz and 1200 MHz operating frequencies
- FSB frequency of 400 MHz and 533 MHz.
- Uni-directional FSB interface (on-chip) for SoC applications
- 32 KByte L1 split instruction and data caches
- 256 KByte 2-way L2 cache with 64B lines
- Soft error protection on L2 cache data and tags (via ECC), soft error protection on L1 cache data and tags (via parity)
- New CPU identifier (CPUID)

Note the differences between the IA-32 core and the Intel® Pentium® M processor (90 nm) are:

- Core and FSB operating frequencies
- Uni-directional FSB instead of bi-directional
- Different CPU identifier (CPUID)
- Reduced L2 cache size and ways
- De-featured Intel SpeedStep® Technology (no VRM specification required)

8.2 Theory of Operation

This section discusses several operational areas of IA-32 core.

8.2.1 L2 Cache Size

The IA-32 core reduces the size and associativity of the L2 cache on the Pentium M processor (90 nm)from 2MB 8-way to 256KB 2-way.

8.2.2 Platform and JTAG Identifiers

The Platform ID is a project-specific value. It is used solely for selecting a microcode update (i.e., patch) and is tightly-coupled to a specific CPUID number. BIOS and OS software are aware of the Platform ID convention and automatically pick up the appropriate patch. There are two views to the Platform ID: the MSR view and the Patch view: The MSR view is a 3-bit value read out of the processor's FUSE_MSR[52:50]. This value is determined by three fuses and is unique per CPUID SKU. The Patch view is the binary value which represents two raised to the power of the MSR view value. A microcode update is considered applicable to a specific processor if, and only if, the following condition is true:

```
(((Patch.header.cpuid == Processor's CPUID) &&

((Patch.header.platformID &  2^FUSE_MSR[52:50]) != 0))
```

The MSR view of the EP80579 Platform ID is 100b. Table 8-1 summarizes the format of the processor version identification signature (CPUID).

Table 8-1. Processor Version Identification Signature (CPUID)

31 28	27 20	19 16	15 14 13 12	11 8	7 4	3 0
	Extended Family	Extended Model	Type	Family	Model Number	Stepping ID
0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 1	0 0	0 1 1 0	0 1 0 1	0 0 0 0

8.2.3 FSB Physical Interface

The IA-32 core replaces the standard bi-directional, tri-state FSB interface found on the Pentium M processor (90nm) with a uni-directional interface. This uni-directional interface is better suited for SoC applications that use a single core. This modification replaces a single bi-directional wire with two uni-directional wires.

8.2.4 IA-32 Core and FSB Frequency

The EP80579 is targeted to operate with IA-32 core frequencies of 600 MHz, 1066 MHz and 1200 MHz and FSB frequencies of 400 MHz (600 MHz SKU) and 533 MHz (1066 MHz and 1200 MHz SKUs).

§ §

9.0 CMI Introduction

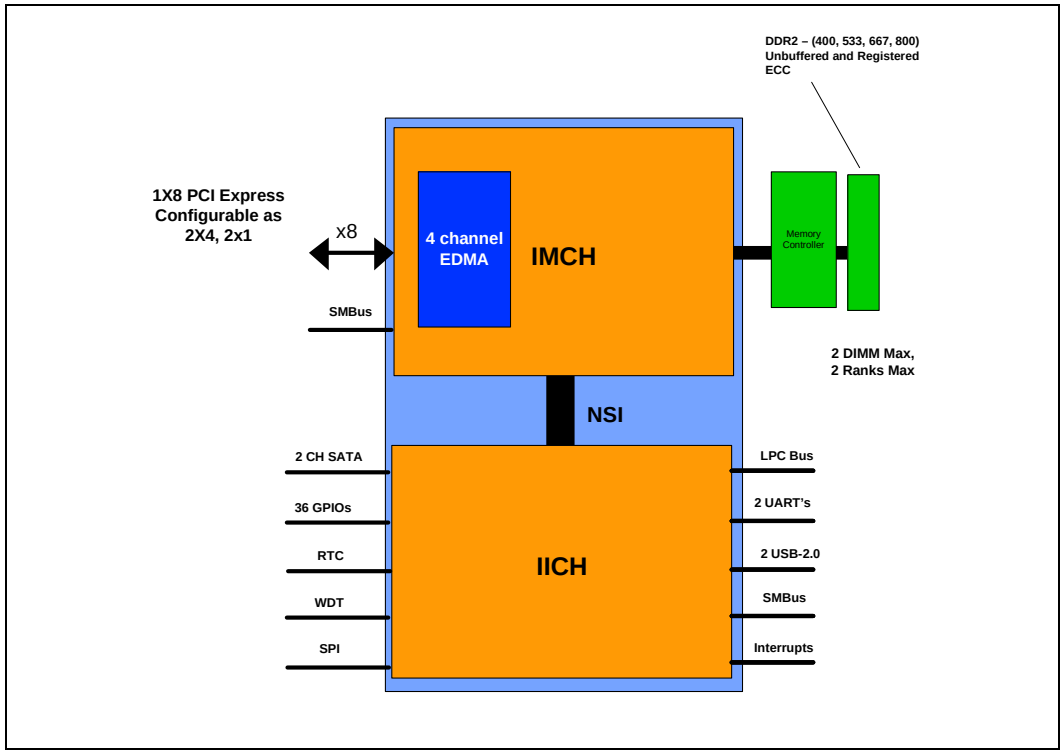
This section details the system architecture supported by the Memory Controller Hub and I/O Controller Hub complex. The Memory Controller Hub and I/O Controller Hub are referred to collectively as the CMI (IMCH and ICH).

Subsequent chapters cover the following aspects of the MCH and the ICH:

- Description of the CMI architecture.
- Descriptions of internal registers.
- Descriptions of all external interfaces.

The EP80579 is a single chip that integrates the functionality of an IA-32 core, Memory Controller Hub, and an I/O Controller Hub (see Figure 9-1). In this document the Memory Controller Hub and I/O Controller Hub in CMI are referred to as IMCH (Integrated Memory Controller Hub) and IICH (Integrated I/O Controller Hub) respectively. The IMCH and IICH units are connected internally through the NSI (North South Interface). The NSI is an internal bus that is not externally accessible.

Figure 9-1. CMI Block Diagram



9.1 System Architecture

CMI implements numerous RASUM (Reliability, Availability, Serviceability, Usability and Manageability) features on multiple interfaces.

The IMCH and IICH consist of:

- A Memory Controller.
- A four-channel, descriptor-chain-based Enhanced DMA (EDMA) controller.
- Several I/O devices such as USB, SATA, etc.
- One x8 PCI Express* interface, which may be split into a pair of independent x4 or x1 PCI Express* interfaces.

Desired I/O Controller Hub (IICH) functions are integrated eliminating the requirement for a legacy I/O bridge.

CMI also supports:

- Two USB 2.0/1.1 ports
- Two SATA ports Gen1/Gen2
- One LPC bus
- One SPI port
- Two UART port
- Two SMBus ports

For additional information see Section 11.3, “Configurations” on page 291.

9.2 PCI Express*

CMI provides one configurable x8 PCI Express interface with a maximum theoretical bandwidth of 4 GByte/s (aggregate). The x8 PCI Express interface may alternatively be configured as two independent x4 or x1 PCI Express interfaces.

CMI is a root-class component as defined in the *PCI Express* Interface Specification, Rev 1.0a*. The PCI Express interfaces support connection of CMI to a variety of other bridges compliant with the same revision of the *PCI Express* Interface Specification, Rev 1.0a*. For example, the Intel® 82571EB Gigabit Ethernet adaptor and the Intel PCI Express I/O processor are directly supported on any of these PCI Express ports. Other compatible PCI Express devices implement functionality such as graphics, hardware RAID controllers and TCP/IP off-load engines. These devices are available from Intel and/or third-party vendors.

As required by the interface specification, CMI will automatically negotiate for and train a single lane (x1) link if an attached device on any logical port fails to establish a viable x4 or x8 connection. This does not imply a capability for CMI to support more than two independent PCI Express ports of any width simultaneously on the x8 port, nor does it imply that the remaining three lanes of a potential x4 port are useful once the associated link has been established for x1 operation. Similarly, CMI will automatically negotiate for and train a single lane (x1) link if an attached device on any logical port fails to establish a viable x4 connection.

External bridge devices such as PCI or PCI-X Gigabit Ethernet or RAID storage devices are directly supported on the PCI Express ports. This does not preclude connection of the IMCH to other bridges compliant with the same revision of the *PCI Express Interface Specification*.

9.2.1 Supported PCI Express Configurations

CMI PCI Express ports are setup using a configuration register. This register is "DEVPRES" BDF 000, Offset 9Ch, Bits 2 and 3 which control Device 2 and Device 3 present.

Table 9-1 shows all of the valid combinations of this register.

Table 9-1. Supported PCI Express Configurations

		DEVPRES			
		DEV2 En, DEV3 En	**DEV2 En, DEV3 Dis**	**DEV2 Dis, DEV3 En**	**DEV2 Dis, DEV3 Dis**
Strap	AUTO Negotiate	1x8, 2x4, lower 1x4, upper 1x4	1x8, lower 1x4	Not supported	All Disabled
	2x4	2x4	lower 1x4	upper 1x4	All Disabled
	1x8	1x8	1x8	Not supported	All Disabled

9.2.1.1 Low power SKU with PCI Express ports removed

To enable low power SKU for configurations that do not utilize the PCI Express ports, the user can disable PCI Express ports and configuration bits for power savings.

9.3 Supported Debug and Management Interfaces

The IMCH supports a target SMBus interface for access and control of the IMCH through its configuration registers. A Test Access Port (TAP) interface is also supported for IMCH system debug purposes. The TAP is capable of full read/write access to the entire internal IMCH register space.

Platforms based on CMI may also make use of the Inter-Chassis Management Bus (ICMB) architecture to extend SMBus based management throughout a desegregate multi-chassis platform solution.

9.4 Supported IMCH Integrated Features

This section provides a brief overview of internal IMCH features. The subsections are intended for use as an introduction and a quick reference. See Section 9.5 for more detailed descriptions of these features.

9.4.1 EDMA Controller

The IMCH includes an integrated four-channel Enhanced Direct Memory Access (EDMA) controller to perform background data transfers between locations in main memory, or from main memory to a memory-mapped I/O destination. These transfers may be

individually designated to be coherent (snooped on the FSB) or non-coherent (not snooped on the FSB), providing improvements in system performance and utilization when cache coherence is managed by software rather than hardware.

Each of the four channels implements an independent set of configuration and status registers, and is capable of fully independent operation. Each channel may operate in a single block transfer mode, or a hardware traversed linked-list scatter/gather mode.

The internal EDMA controller only supports transfers between main memory locations, and transfers from a main memory source to an I/O subsystem destination. The internal EDMA controller supports neither transfers between I/O interfaces, nor transfers from an I/O interface source to a main memory destination.

9.4.2 Integrated Memory Init/Test Engine

The IMCH provides hardware-managed ECC memory auto-initialization and testing of all populated DRAM space under software control. Once internal configuration has been updated to reflect the type and size of populated DIMM, the IMCH can traverse the populated address space issuing line-sized writes of all zero data, thereby initializing all locations with good ECC memory. This greatly speeds up the mandatory memory initialization step and frees the CPU to pursue other machine initialization and configuration tasks.

Additional features have been added to the initialization engine to support high-speed population and verification of a programmable memory range with one of eight known data patterns, random data, a walking data pattern, or an explicitly specified cache line (data plus ECC). This function facilitates a limited high-speed memory test and provides BIOS-accessible memory testing capability for potential use by management code or by the operating system.

For additional information see Section 11.2, “Memory Controller Feature List” on page 289.

9.4.3 Coherent Memory Write Buffer

The IMCH includes an integrated coherent write buffer sized for 16 64-byte cache lines (a total of 1 Kbyte of storage). This feature enables the IMCH to optimize memory read latency, allowing reads to pass less critical writes en-route to the main memory store. The write buffer includes a CAM structure to enforce ordering among conflicting accesses to the same cache line, as well as to provide for read service from the write cache. In the latter case, the access to the main memory store never occurs, which both improves latency and conserves bandwidth on the memory interface.

The write buffer is capable of servicing processor read requests directly via a “hit” to the internal location containing the data without initiation of any DDR subsystem accesses. Inbound read requests such as PCI Express, i.e. not processor, which “hit” the write buffer result in a flush of the target data, followed by retrieval via an external read request.

Processor writes to shared non-coherent address space with the ASU result in a flush of the current cacheline to main memory. ASU atomics will result in a DW write to the ASU out of the write cache.

9.4.4 RASUM Features

The IMCH is designed to bring enterprise-level reliability, availability, serviceability, usability, and manageability to the embedded platform. All internal SRAM memory arrays are covered by parity. CMI’s PCI Express interface supports detection and automatic recovery for all transient signaling errors. All IMCH internal configuration

register space is accessible from the system management bus (SMBus) to facilitate system management. The IMCH supports ACPI power management, PCI Express native hot-plug, and wake-from-LAN to maximize platform stand-by flexibility.

9.4.4.1 SEC-DED ECC

The IMCH supports a standard (72-bit, non-interleaved) single error correction (SEC) and double error detection (DED) ECC mechanism for the DDR memory.

The IMCH supports both ECC and non-ECC DIMMs.

For additional information see Section 11.2, “Memory Controller Feature List” on page 289.

9.4.4.2 Integrated Memory Scrub Engine

The IMCH includes an integrated engine to walk the populated memory space proactively seeking out soft errors in the memory subsystem. This hardware detects, logs, and corrects any single-bit ECC errors it encounters, and logs any uncorrectable errors it encounters. Both types of errors may be reported via multiple alternate mechanisms under configuration control. The scrub hardware will also execute “demand scrub” writes when correctable errors are encountered during normal operation (on demand reads, rather than scrub-initiated reads). This functionality provides incremental protection against time-based deterioration of soft memory errors from correctable to uncorrectable.

An uncorrectable error encountered by the memory scrub engine is a “speculative error.” This designation is applied because no system agent has specifically requested use of the corrupt data, and no real error condition exists in the system until that occurs. It is possible that the error resides in an unmodified page of memory that is simply dropped on a swap back to disk. If that were to occur, the speculative error would simply “vanish” from the system without any adverse consequences.

9.5 IMCH Feature List

This section provides an overview of the major IMCH architectural features. Detailed usage information and operational flows, internal register bit information and other specific details of the implementation are provided later in this document.

9.5.1 Memory Interface

For additional information see Section 11.2, “Memory Controller Feature List” on page 289.

9.5.2 PCI Express Interface in IMCH

- Support for one x8 PCI Express dual-simplex, high-speed serial I/O interface with eight striped differential pairs in each direction (outbound and inbound)
  - — The interface may be unpopulated; connected to PCI, Ethernet, I/O Processor, Infiniband* bridge devices, External bridge devices (PCI or PCI-X Gigabit Ethernet or RAID storage devices); or connected to any other device compliant with the same revision of the *PCI Express Specification* as CMI.
  - — The x8 interface is capable of bifurcation into two logically independent x4 interfaces with full specification compliance at half the bandwidth capability
- This interface is referred to throughout this document as the PCI Express Port A (PEA). When configured as x8, the reference is PEA. When in x4 mode there are two available x4 ports referred to as PEA0 and PEA1.

- Raw bit-rate on the data pins of 2.5 Gbit/s.
  - Maximum theoretical realized bandwidth on the x8 PCI Express interface of 2 GByte/s in each direction simultaneously, for an aggregate of 4 GByte/s.
  - Maximum theoretical realized bandwidth on the x4 PCI Express interface of 1 GByte/s in each direction simultaneously, for an aggregate of 2 GByte/s per port.
  - x8 sustainable data bandwidth is approximately 1.6 GByte/s in each direction simultaneously
- Plesiochronous operation with automatic clock extraction and phase correction at the receiver.
- Hierarchical PCI-compliant configuration mechanism for downstream devices
  - Support for PCI Express memory-mapped enhanced configuration mechanism, up to 4 Kbyte per device.
- 64 bit addressing support.
  - 64 bit upstream addressing (full DAC support), limited to 32 bits internally to/ from system memory (external DDR).
  - 32 bit downstream addressing support.
  - Full 36 bit support for peer segment accesses.

Note: Only 32-bit addresses can be snooped. Addresses larger than 32 bits will be truncated. e.g. If a 36-bit address is snooped the upper 4 bits are ignored.

- Full-speed interface self-test and diagnostic (IBIST) functionality.
- Automatic discovery, negotiation, and training of PCI Express ports out of reset.
  - Automatic detection of widest operational link; x8, x4 or x1.
- No support for Hot-plug via an external SMBus connected device.
- Run-time detection and recovery for loss of link synchronization.
- 32 bit CRC (cyclic redundancy checking) on all transaction layer packets with link-level retry on error (recovery from transient errors without software-visible system failure).
- 16 bit CRC on all link message information
- No support for the optional extended CRC (ECRC) mechanism
- Aggressive transceiver design to facilitate flexible system topologies
- Target BER of 10^{-12} for physical signaling interface
- Support for peer segment destination write traffic (no peer-to-peer read traffic)
- Support for coherent and non-coherent transactions through EDMA to PEA to external agent
- Support for both coherent and non-coherent traffic to memory within VC#0
  - Non-coherent implies a combination of Snoop-Not-Required and Relaxed-Ordering attributes
  - Coherent traffic implies a combination of Snoop-Required and Strong-Ordering attributes.
- Support for lane reversal at all native widths, and for reversed x4 training on any x8 port
- Support for peer segment PCI interrupt forwarding to the IICH for boot from I/O
  - Legacy mode support for level-sensitive interrupt emulation without IOxAPIC support

- APIC and MSI interrupt messaging support DMA internal score-boarding to translate messages into level-sensitive IICH pin semantics
  - — XTPR-based interrupt redirection for APIC messages with lowest priority tie breaking
- Support for up to 256B read completion combining
- Support for link messaging to facilitate active link and device power state management
- Support for ASPM L0s entry, no support for optional L1 ASPM support
- No support for inbound configuration or I/O traffic
- No support for inbound special cycles or writes requiring completions
- No support for downstream special cycle messages requiring completions

9.5.3 EDMA Controller

- Four Independent Channels
  - — Dedicated data transfer queue per channel
  - — Full register set for descriptor and transfer handling per channel
- Support for transfer between main memory locations, and from memory to the I/O subsystem
- Supports PCI Express traffic class to allow external prioritization of traffic
- Supports transfers only between two Physical Addresses
  - — 32 bit (4 GB) addressing range on the Local System Memory Interface
  - — 32 bit addressing range on the Memory Mapped I/O Subsystem Interface
- Maximum transfer of 16 Mbyte transfers per block
- Fully programmable by the host CPU
  - — Configuration space mapping for DMA engine capability and control
  - — Memory-mapped space for DMA channel-specific register sets
- Chain mode DMA transfer with automatic data chaining for scattering/gathering of data blocks
  - — DMA chaining continued until a “null” Descriptor Pointer is encountered
  - — Support for appending a block to the end of current DMA chain
  - — Automated descriptor retrieval from DDR during chaining – single read
- Programmable independent alignment between source and destination
  - — Byte aligned transfer on the DDR Memory Interface
  - — Byte aligned transfer on the I/O Subsystem Interface
- Support for non-coherent transfers both to and from system memory on a per descriptor basis
  - — Independent control of coherency for source and destination
- Programmable support for interrupt generation on block–by-block basis
  - — Selectable MSI or legacy level-sensitive interrupt function
  - — End of current block transfer
  - — End of current chain
  - — For any error causing a transfer to abort

- Increment of the source and destination address for standard transfers
- Increment of the destination and decrement of the source address to enable byte stream reversal.
- Constant address mode for the destination address based on the transfer granularity to enable targeting of memory mapped I/O FIFO devices
- Buffer/Memory Initialization Mode

9.5.4 Coherent Memory Write Buffer

- Support for 16 64-byte cache-lines of write data
- Fully associative conflict detection for accesses targeting memory
- Read around write support (non-conflicting) for all traffic to memory
- Read-hit support for CPU traffic to memory
  - — Direct data service from buffer without generation of memory traffic
- Write-hit support for memory traffic with address conflicts
  - — Hardware-based merging to collapse down to a single memory write
- Opportunistic and demand (buffer full) mode processing of pending writes
  - — Configurable “watermark” mechanism for hardware-based prioritization
  - — Flush on demand via software configuration mechanism
- Parity protection on all data
- Data poisoning capability in the main store for data received with errors
- Processor writes to shared non-coherent address space with the ASU result in a flush of the current cacheline to main memory
- ASU atomics will result in a DW write to the ASU out of the write cache

9.5.5 Integrated Memory Scrub Engine

For additional information see Section 11.2, “Memory Controller Feature List” on page 289.

- Periodic (programmable) read-modify-write algorithm
- Support for the SEC-DED mode of operation
- Automatic correction of encountered SEC errors
- Logging of detected errors with granularity to isolate DRAM device
  - — Support for logging of both first and next subsequent error
  - — Count of errors beyond the first two which are logged
- Support for on-demand hardware scrub of SEC errors detected during normal operation
- Programming interface permits software suspend/resume of scrub in progress

9.5.6 Hardware Memory Initialization Engine

- Available via BIOS for hardware memory initialization and/or test
- Provides fast WHQL initialization of all populated DRAM space to “0” with good ECC
- Target region may be a single location, an entire rank, or all populated ranks
- Algorithm optimized for speed, runs at DDR channel saturation rate

- Test extensions permit high-speed population of a target range with a known pattern
  - — Selectable hardware-generated fixed patterns: 0, 3, 5, 6, 9, A, C, F
  - — Hardware generated random pattern capability
  - — Explicitly stipulated data pattern including ECC
- High speed verification capability
  - — Selectable write-only, verify-only, or write-read-verify per location
  - — Logs error location, optional stop and escalate on error detection
- May be made available to the operating system via BIOS for security “clear to 0” function

9.5.7 System Management Functions

- Full SMBus target support
- Support for remote chassis management via the ICMB architecture
- Serial presence detect of memory devices via standard I^2C protocol (accessed via IICH)
- ACPI and PCI-PM compatible power management
  - — Includes PME support comprehending PCI Express extensions
- MSI interrupt messaging and redirection support
- Hardware relay of PCI Express legacy mode PCI interrupt messages to IICH
  - — Supports boot from I/O when IOxAPIC functions are unavailable

9.5.8 RASUM

- SEC/DED ECC protection of external memory DRAM data
- Parity protection on internal data propagated through the IMCH
- CRC on data packets and hardware link-level retry on NSI to the IICH
- 32-bit CRC on data packets and hardware link-level retry on PCI Express ports
- Hardware memory initialization
  - — True “clear-to-zero” via hardware writes to all populated devices
  - — Support for hardware-based fast initialization of memory with selectable patterns
  - — Support for hardware-based fast verification of memory via accelerated scrub
- Hardware periodic memory scrubbing, including demand scrub support
- Flexible extended error reporting capabilities
- Configurable error containment at I/O interfaces (poison/propagate or stop/escalate)
- Partial access to internal configuration registers via the TAP port
- Full access to internal configuration registers via SMBus port

9.6 IICH Feature List

This section provides a listing of architectural functionality for the major features of the IICH. Detailed usage information and operational flows, internal register bit information, and other specific details of the implementation are provided later in this document.

9.6.1 Low-Pin count (LPC) Interface and Firmware Hub (FWH) Interface

- Allows connection of devices such as Super I/O, micro controllers, customer ASICs.
- Supports two master/DMA devices.
- Memory size up to 8 Mbit.

9.6.2 Serial Peripheral Interface (SPI)

Note: Intel recommends using the SPI as your boot interface.

- Supports multiple SPI Flash vendors.
- Simplified Hardware.
- Equivalent to LPC-based Firmware Hubs.

9.6.3 Integrated Serial ATA Host Controllers

- Independent DMA operation on two ports.
  - — Two ports in SATA 1.0a and AHCI mode.
  - — Two ports in AHCI mode only.
- Data transfer rates up to 300 Mbyte/s.
- Support Gen2m electrical spec (cable not exceeding 2m).

9.6.4 USB

- One EHCI USB 2.0 Host Controller with a total of two ports (shared with the UHCI ports).
- One UHCI Host Controller for a total of two ports (shared with the EHCI ports).
- Supports a Debug Port at USB 2.0 transfer rates.

9.6.5 Interrupt Controller

- Supports up to 8 PCI interrupt pins.
- Two cascaded 82C59 with 15 interrupts.
- Integrated I/O APIC supports a total of 40 interrupts (24 interrupts only, when ETR3.GPIO_IRQ_STRAP_STS is 0).
- Serial Interrupt input for ISA legacy-compatible and PCI interrupts.
- Supports PCI scheme for delivering interrupts as write cycles (rather than via PIRQ[A-H]#).
- Front-Side Message Interrupt Delivery.
- Supports EOI message.

9.6.6 Power Management Logic

Power management logic for various environments require updates.

- ACPI 2.0 Compliant.
- Support for APM-based legacy power management for non-ACPI implementations.
- Supports ACPI defined power states S0, S1, S3 cold, S4 and S5 (SOFF).
- ACPI Power Management Timer.
- SMI# Generation.
- PCI PME#.

9.6.7 DMA Controller

- Two cascaded 8237 DMA Controllers.
- Supports LPC DMA.

9.6.8 Timers Based on 82C54

- System Timer, Refresh Request, Speaker Tone Output.

9.6.9 High Precision Event Timers (HPET)

- Three timer comparators provided.
- One-shot and periodic interrupts supported.

9.6.10 Real-Time Clock with 256-byte Battery-backed CMOS RAM

- Integrated components for the oscillator to reduce problems with incorrect external selections.
- Lower Power DC/DC Converter implementation.

9.6.11 System TCO Reduction Circuits

- Timers to detect improper CPU reset and to generate SMI# and Reset upon detection of stuck CPU.
- Interrupt capability to OS-specific manageability extension and OS capability to call TCO BIOS.
- Supports CPU BIST.
- Ability to disable external devices.

9.6.12 SMBus

- Host interface allows CPU to communicate via SMBus.
- Compatible with most 2-wire components that are also I^2C compatible.
- Slave interface allows internal or external microcontroller to access system resources.
- SMBus 2.0 Compliant.
- No ASF support.

9.6.13 Watchdog Timer

- Selectable Prescaler:
  - Approximately 1 MHz (1 µs to 1 s).
  - Approximately 1 KHz (1 ms to 10 min.)
- 33 MHz Clock (30 ns Clock Ticks).
- Multiple Modes (WDT and Free-Running).
- Free-Running Mode:
  - One Stage Timer.
  - Toggles WDT_TOUT# after programmable time.
- WDT Mode:
  - Two Stage Timer (First Stage generates Interrupt, Second Stage drives WDT_TOUT# low).
  - First Stage generates an SERIRQ, NMI or SMI interrupt after Programmable time.
  - Second Stage drives WDT_OUT# low or inverts the previous value.
  - Used only after first timeout occurs.
  - Status bit preserved in RTC well for possible error detection and correction.
  - Drives WDT_TOUT# if OUTPUT is enabled.
- Timer can be disabled (default state) or Locked (Hard Reset required to disable WDT).
- WDT Automatic Reload of Preload value when WDT Reload Sequence is performed.

9.6.14 Serial Port

- Two Full Function 16550 Compatible Serial Ports.
- Configurable I/O addresses and interrupts.
- 16-Byte FIFOs.
- Supports up to 115 Kbps.
- Programmable Baud Rate Generator.
- Modem Control Circuitry.
- 14.7456 MHz and 48 MHz supported for UART baud clock input.

9.6.15 GPIO

- General Purpose I/Os .
- There are 36 GPIO pins of which 5 have alternate power on functions.

§ §

10.0 System Address Map

10.1 Overview

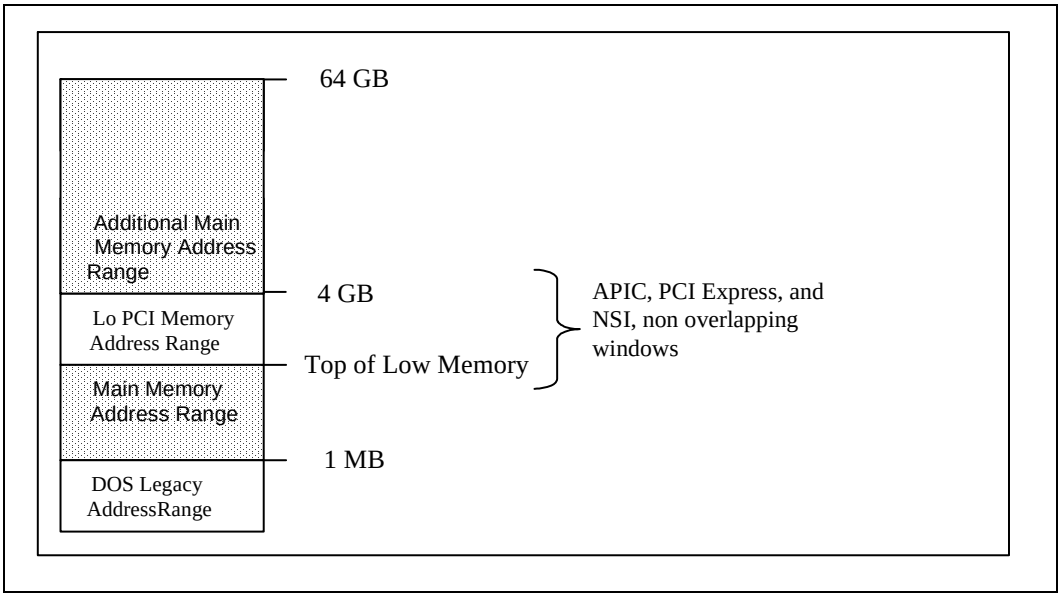
The IA-32 core addressable memory map is 32 bits or up to 4 GBytes and has 64 KBytes+3 of addressable I/O space. The I/O and memory spaces are divided by system configuration software into non-overlapping regions. The memory ranges are useful either as system memory or as specialized memory, while the I/O regions are used solely to control the operation of devices in the system.

There are five basic regions of memory in the system. The regions are shown in Table 10-1. Figure 10-1 illustrates the basic memory regions.

Table 10-1. Regions of Memory Ranges

Range	Description
Between top of main memory and 64 GBytes	High PCI Memory Range
Between 4 GBytes and top of main memory	
Between TOLM Register and 4 GBytes	Low PCI Memory Address Range
Between 1 MByte and the TOLM Register	Main Memory Address Range
Below 1 MByte	DOS Legacy Address Range

Figure 10-1. Basic Memory Regions



10.1.1 System Memory Spaces

Table 10-2. System Memory Space

	From	To
DOSMEM	0_0000_0000	0_0009_FFFF
MEM1_15	0_0010_0000	0_00EF_FFFF
MAINMEM	0_0100_0000	TOLM
HIGHMEM	1_0000_0000	7_FFFF_FFFF

Table 10-2's address ranges are always mapped to system memory, regardless of the system configuration. The Top of Low Memory (TOLM) register (see Section 16.1.1.30, "Offset C4h: TOLM - Top of Low Memory Register") provides a mechanism to carve memory out of the MAINMEM segment for use by System Management Mode (SMM) hardware and software, PCI add-in devices, and other functions. The address of the highest 128 MByte quantity of populated DRAM memory in the system is placed into the DRB3 register, which will match the value in the Top of Memory (TOM) register (see Section 16.1.1.34, "Offset CCh: TOM - Top Of Memory Register".

10.1.2 VGA and MDA Memory Spaces

Table 10-3 lists the VGA and MDA Memory spaces. Figure 10-2 illustrates the DOS legacy Region.

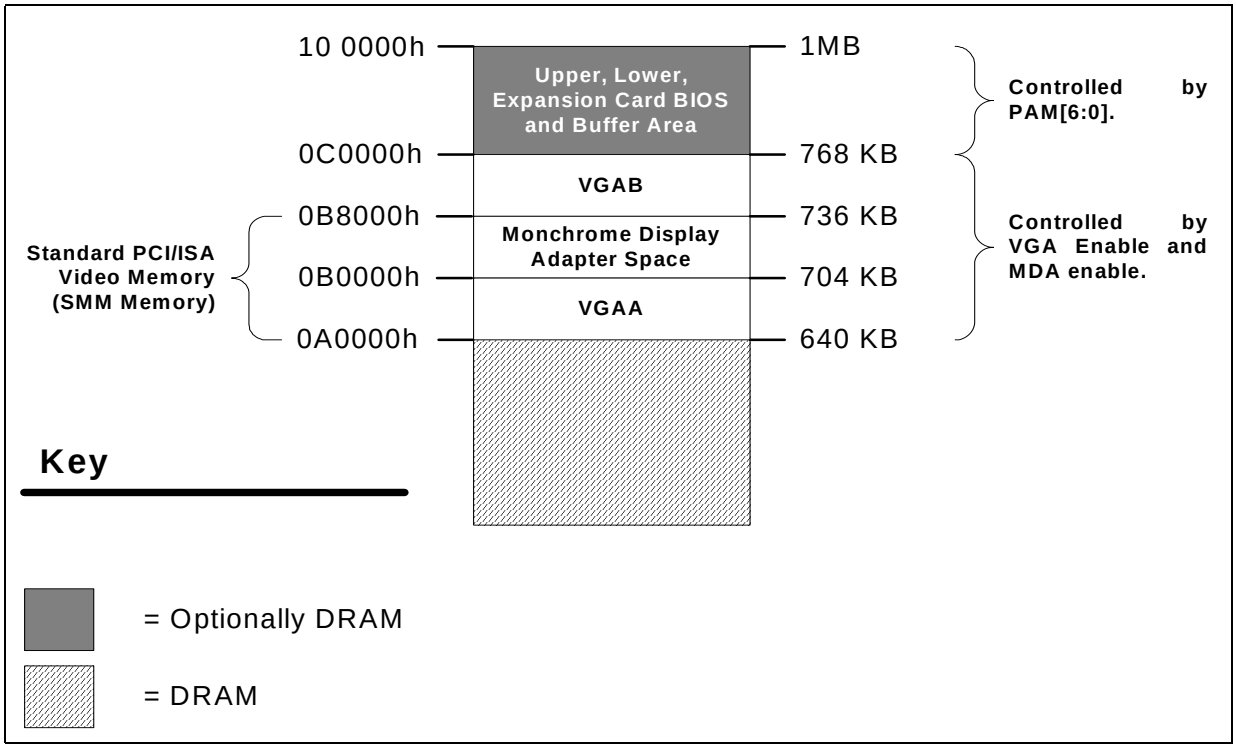
Table 10-3. IMCH VGA and MDA Memory Spaces

	From	To
VGAA	0_000A_0000	0_000A_FFFF
MDA	0_000B_0000	0_000B_7FFF
VGAB	0_000B_8000	0_000B_FFFF

These legacy address ranges are used on behalf of video cards to map a frame buffer or a character-based video buffer into a dedicated location. By default, accesses to these ranges are forwarded to the NSI. However, if the VGAEN bit is set in one of the BCTRL configuration registers (see Section 16.4.1.26, "Offset 3Eh: BCTRL - Bridge Control Register"), then transactions within the VGA and MDA spaces are sent to one of the PCI Express interfaces in IMCH.

Note: The VGAEN bit may be set in *one and only one* of the BCTRL registers. Software *must not* set more than one VGAEN bit.

Figure 10-2. DOS Legacy Region



If the configuration bit EXSMRC.MDAP (see Section 16.1.1.25, “Offset 9Dh: EXSMRC - Extended System Management RAM Control Register”) is set, then accesses that fall within the MDA range are sent to NSI without regard for the VGAEN bits. Legacy support requires the ability to have a second graphics controller (monochrome display adapter) in the system. In a CMI system with PCI graphics installed via a PCIe to PCI bridge like PXH, accesses in the standard VGA range may be forwarded to any of the logical PCI Express ports (depending on configuration bits). Since the monochrome adapter may be on the NSI (or logical ISA) bus, the IMCH must decode cycles in the MDA range and forward them to NSI. This capability is controlled via the MDAP configuration bit. In addition to the memory range B0000h to B7FFFh, the IMCH decodes I/O cycles at 3B4h, 3B5h, 3B8h, 3B9h, 3BAh and 3BFh and forwards them to NSI.

An optimization allows the system to reclaim the memory displaced by these regions. If SMM memory space is enabled by EXSMRC.G_SMRAME and either the SMRAM.D_OPEN bit (see Section 16.1.1.25, “Offset 9Dh: EXSMRC - Extended System Management RAM Control Register” and Section 16.1.1.26, “Offset 9Eh: SMRAM - System Management RAM Control Register”) is set or the processor bus receives an SMM-encoded request for code (not data), then the transaction is steered to system memory rather than NSI. Under these conditions, both the VGAEN bits and the MDAP bit are overridden.

If any VGAEN bit is set, then all ISAEN bits (see Section 16.4.1.26, “Offset 3Eh: BCTRL - Bridge Control Register”) must be set. The *PCI Specification* defines VGAEN to be 10-bit decode. Therefore the other peer bridges must also be 10-bit decodes (ISAEN), so that two or more devices don't claim same access. (Bridge C doesn't know bridge B has its VGAEN bit set.)

The MDA bit may only be set when one of the VGAEN bits is set. If no VGAEN bit is set, then MDA must not be set either. When the VGA range is already mapped onto the NSI interface, the MDA range is included as a subset, and the MDA enable is meaningless.

10.1.3 PAM Memory Spaces

The Address range for the PAM memory space is defined in Table 10-4.

Table 10-4. IMCH PAM Memory Address Ranges

	From	To	Access Region
PAMC0	0_000C_0000	0_000C_3FFF	ISA Expansion (16KB/each)
PAMC4	0_000C_4000	0_000C_7FFF	
PAMC8	0_000C_8000	0_000C_BFFF	
PAMCC	0_000C_C000	0_000C_FFFF	
PAMD0	0_000D_0000	0_000D_3FFF	
PAMD4	0_000D_4000	0_000D_7FFF	
PAMD8	0_000D_8000	0_000D_BFFF	
PAMDC	0_000D_C000	0_000D_FFFF	
PAME0	0_000E_0000	0_000E_3FFF	Extended BIOS (16KB/each)
PAME4	0_000E_4000	0_000E_7FFF	
PAME8	0_000E_8000	0_000E_BFFF	
PAMEC	0_000E_C000	0_000E_FFFF	
PAMF0	0_000F_0000	0_000F_FFFF	System BIOS (64KB)

The 256 Kbyte Programmable Access Memory (PAM) region is divided into three parts:

- ISA expansion region, a 128 Kbyte area between 0_000C_0000h – 0_000D_FFFFh.
- Extended BIOS region, a 64 Kbyte area between 0_000E_0000h – 0_000E_FFFFh.
- System BIOS region, a 64 Kbyte area between 0_000F_0000h – 0_000F_FFFFh.

Specialized programmable hardware in the IMCH supports routing of read and write accesses within the PAM region independently to memory or to NSI.

Non-snooped transactions are treated accordingly:

- Non-snoop Reads: Memory address 0h. The result is an unsupported request (UR) completion.
- Non-snoop Writes: Memory address 0h with byte enables deasserted.

The ISA expansion region is divided into eight 16 Kbyte segments. Each segment can be assigned one of four Read/Write memory states: read-only, write-only, read/write, or disabled. These segments are typically set to disabled for memory access, which leaves them routed to NSI for ISA space.

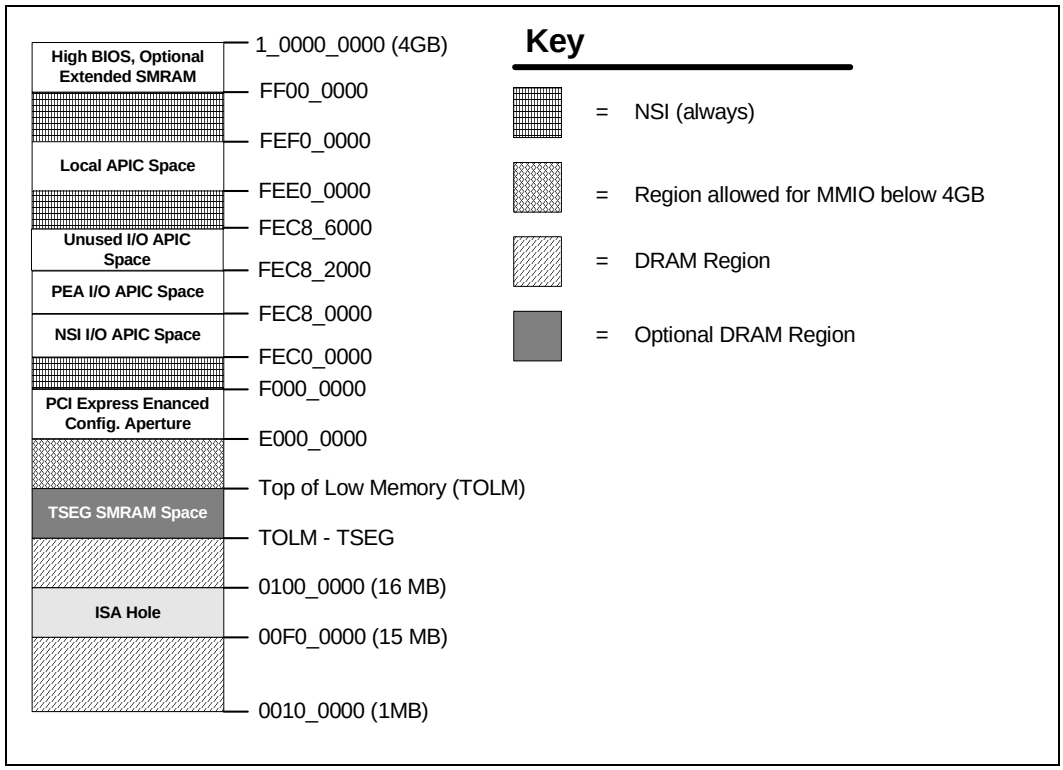
The extended System BIOS region is divided into four 16 Kbyte segments. Each segment can be assigned independent read and write attributes so it can be mapped either to main DRAM or to NSI. Typically, this area is used for RAM or ROM.

The system BIOS region is a single 64 Kbyte segment. This segment can be assigned independent memory read and write attributes. It is by default (after reset) Read/Write disabled and cycles are forwarded to NSI. By manipulating the Read/Write attributes, the IMCH can "shadow" BIOS into the main DRAM. The term "shadow" is used to describe the condition where ROM memory has been duplicated into main memory; such that reads are serviced from memory, while writes are directed back to the original ROM device. Such a configuration allows low-latency reads of BIOS information from the ROM while preventing malicious or inadvertent alteration of the BIOS information in use.

Note: The PAM regions are generally inaccessible from the logical PCI Express ports. All inbound writes from any port that hit the PAM regions are sent to NSI, which prevents corruption of non-volatile data shadowed in main memory. All inbound reads from any port that hit the PAM regions are harmlessly terminated internally; data is returned, but not necessarily from the requested address. Transaction routing is not hardware enforced based on the settings in the PAM configuration registers.

Note: The PAM regions are inaccessible from the logical AIOC port. All inbound reads/writes from any port that hit the PAM regions are master aborted by the AIOC preventing them from ever pushed into the IMCH.

Figure 10-3. Memory Region from 1 MByte through 4 GBytes



The IMCH allows programmable memory attributes on 13 legacy memory segments of various sizes in the 768 Kbyte to 1 Mbyte (C0000h – FFFFFh) and 640 Kbytes to 1 Mbyte address range. Seven Programmable Attribute Map (PAM) registers are used to support these features. Not all seven of these registers are identical. PAM0 controls only one segment (high), while PAM[1:6] each control two segments (high and low). Cache ability of these areas is controlled via the MTRR registers in the processor. The following two bits apply to both host accesses and PCI initiator accesses to the PAM areas and are used to specify the memory attributes for each memory segment: These bits apply to both host accesses and PCI initiator accesses to the PAM areas

RE — Read Enable. When RE = 1, the IA-32 core read accesses to the corresponding memory segment are claimed by the IMCH and directed to main memory. Conversely, when RE = 0, the host read accesses are directed to the IICH's PCI bus.

WE — Write Enable. When WE = 1, the host write accesses to the corresponding memory segment are claimed by the IMCH and directed to main memory. Conversely, when WE = 0, the host write accesses are directed to the IICH's PCI bus.

Together, these two bits specify memory attributes (Read-Only, Write Only, Read/Write and Disabled) for each memory segment. These bits only apply to host-initiated access to the PAM areas. The IMCH forwards to main memory any PCI Express initiated accesses to the PAM areas. At the time such PCI Express accesses to the PAM region may occur, the targeted PAM segment must be programmed to Read/Write. It is illegal to issue a PCI Express initiated transaction to a PAM region with the associated PAM register not set to Read/Write.

As an example, consider a BIOS that is implemented on the expansion bus. During the initialization process, BIOS can be shadowed to main memory to increase system performance. When BIOS is shadowed to main memory it must be copied to the same address location. To shadow the BIOS, the attributes for that address range must be set to Write-Only. The BIOS is shadowed by first doing a read of that address, which is forwarded to the expansion bus. The host then writes the same address, which is directed to main memory. After BIOS is completely shadowed, the attributes for that memory area are changed to Read-Only so that all writes are forwarded to the expansion bus. Figure 10-4 and Table 10-5 show the PAM registers and the associated attribute bits.

Figure 10-4. PAM Associated Attribute Bits

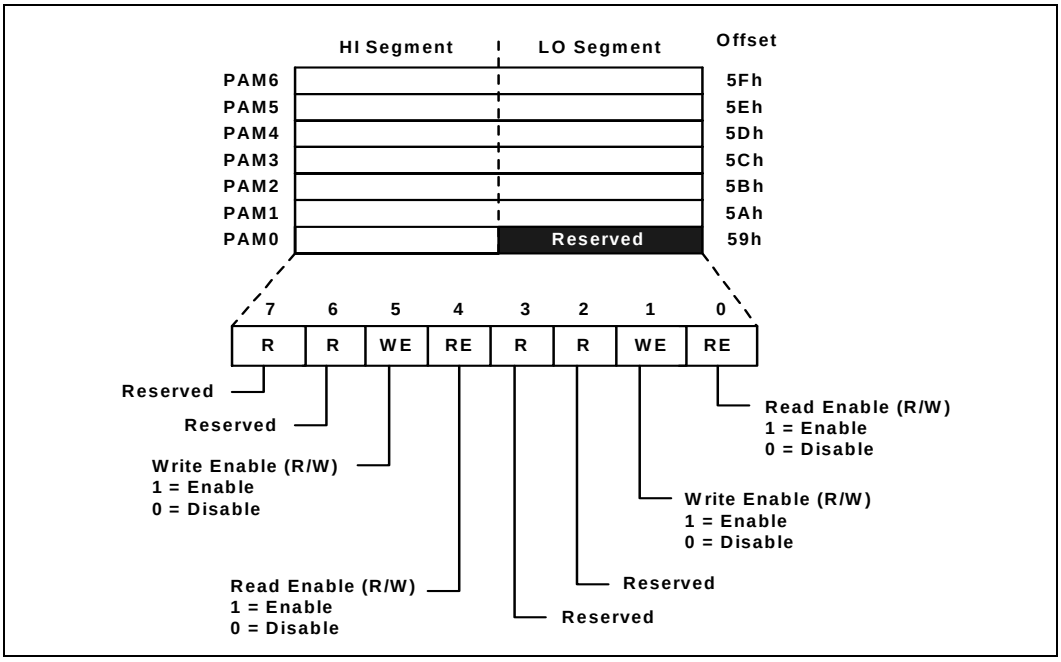


Table 10-5. PAM Associated Attribute Bits

PAM Reg	Attribute Bits		Memory Segment	Comments	D0:F0 Offset
PAM0 03:00, 07:06	Reserved		—	Reserved	59h
PAM0 05:04	WE	RE	0F0000h–0FFFFFh	BIOS Area	59h
PAM1 03:02, 07:06	Reserved		—	Reserved	5Ah
PAM1 01:00	WE	RE	0C0000h–0C3FFFh	BIOS Area	5Ah
PAM1 05:04	WE	RE	0C4000h–0C7FFFh	BIOS Area	5Ah
PAM2 03:02, 07:06	Reserved		—	Reserved	5Bh
PAM2 01:00	WE	RE	0C8000h–0CBFFFh	BIOS Area	5Bh
PAM2 05:04	WE	RE	0CC000h–0CFFFFh	BIOS Area	5Bh
PAM3 03:02, 07:06	Reserved		—	Reserved	5Ch
PAM3 01:00	WE	RE	0D0000h–0D3FFFh	BIOS Area	5Ch
PAM3 05:04	WE	RE	0D4000h–0D7FFFh	BIOS Area	5Ch
PAM4 03:02, 07:06	Reserved		—	Reserved	5Dh
PAM4 01:00	WE	RE	0D8000h–0DBFFFh	BIOS Area	5Dh
PAM4 05:04	WE	RE	0DC000h–0DFFFFh	BIOS Area	5Dh
PAM5 03:02, 07:06	Reserved		—	Reserved	5Eh
PAM5 01:00	WE	RE	0E0000h–0E3FFFh	BIOS Extension	5Eh
PAM5 05:04	WE	RE	0E4000h–0E7FFFh	BIOS Extension	5Eh
PAM6 03:02, 07:06	Reserved		—	Reserved	5Fh
PAM6 01:00	WE	RE	0E8000h–0EBFFFh	BIOS Extension	5Fh
PAM6 05:04	WE	RE	0EC000h–0EFFFFh	BIOS Extension	5Fh

See Section 16.1.1.17, “Offset 59h: PAM0 - Programmable Attribute Map 0 Register” through Section 16.1.1.23, “Offset 5Fh: PAM6 - Programmable Attribute Map 6 Register” for more register information on PAM memory space registers.

10.1.4 TSEG SMM Memory Space

Table 10-6. TSEG SMM Memory Space

	From	To
TSEGSMM	TOLM - TSEG	TOLM

The TSEG SMM space allows system management software to partition a region of main memory just below the top of low memory (TOLM) that is accessible only by system management software.

Size — 128kB, 256kB, 512kB, or 1 MByte in size, depending upon the EXSMRC.TSEG_SZ field (see Section 16.1.1.25).This space must be below 4 GBytes, so it is specified relative to TOLM and not relative to the top of physical memory.

Enabling — SMM memory is globally enabled by EXSMRC.G_SMRAME (see Section 16.1.1.25). Requests may access SMM system memory when either SMM space is open (see SMRAM.D_OPEN in Section 16.1.1.26) or the IMCH receives an SMM code request on its processor bus.

Access — In order to access the TSEG SMM space, the TSEG must be enabled by EXSMRC.T_EN (Section 16.1.1.25). When all of these conditions are met, then a processor bus access to the TSEG space (between TOLM-TSEG and TOLM) is sent to system memory. If the high SMRAM is not enabled or if the TSEG is not enabled, then all memory requests from all interfaces are forwarded to system memory. If the TSEG SMM space is enabled, and an agent attempts a non-SMM access to TSEG space, then the transaction is specially terminated.

Inbound accesses from NSI or PCI Express ports are not allowed to access SMM space.

10.1.5 PCI Express Enhanced Configuration Aperture

Table 10-7. PCI Express Enhanced Configuration Aperture

	From	To
HECREGION	0_E000_0000	0_EFFF_FFFF

PCI Express defines a memory-mapped aperture mechanism through which to access 4 Kbyte of PCI configuration register space for each possible bus, device, and function number. This 4 Kbyte space includes the compatible 256 B of register offsets that are traditionally accessed via the legacy CF8/CFC configuration aperture mechanism in I/O address space, making the enhanced configuration mechanism a full superset of the legacy mechanism. The enhanced mechanism has the advantage that full destination and type of access is specified in a single memory-mapped uncacheable transaction on the FSB, which is both faster and more robust than the historical I/O-mapped address and data register access pair.

CMI places the enhanced configuration aperture at E000_0000h by default, as this is the first contiguous 256 MByte location below the 4 GByte boundary available for such usage.

CMI provides for relocation of this aperture via the HECBASE register (see Section 16.1.1.35, "Offset CEh: HECBASE - PCI Express Port A (PEA) Enhanced Configuration Base Address Register", although validation of moving the region is minimal.

10.1.6 IOAPIC Memory Space

Table 10-8. IOAPIC Memory Space

	From	To
IOAPIC0 (NSI)	0_FEC0_0000	0_FEC7_FFFF
IOAPIC2 (PEA0)	0_FEC8_0000	0_FEC8_0FFF
IOAPIC3 (PEA1)	0_FEC8_1000	0_FEC8_1FFF

The IOAPIC spaces are used to communicate with IOAPIC interrupt controllers that may be populated on NSI through PCI-Express port A (PEA). Since it is difficult to relocate an interrupt controller using plug-and-play software, fixed address decode regions have been allocated for them. Processor accesses to the IOAPIC0 region are always sent to NSI. Processor accesses to the IOAPIC2 region are always sent to PEA. These regions are subject to the APIC disable, which are cleared by BIOS after the allocated regions have been reflected down to the base registers of APIC controllers discovered during standard enumeration. Until this step of the initialization sequence has been performed, accesses to these regions are treated as subtractive decode and routed to NSI.

The IMCH does not support an IOAPIC range for the EDMA controller or the AIOC, since there is no IOxAPIC device or corresponding register set integrated into the EDMA controller or the AIOC.

10.1.7 FSB Interrupt Memory Space

Table 10-9. FSB Interrupt Memory Space

	From	To
FSBINTR	0_FEE0_0000	0_FEEF_FFFF

The FSB Interrupt space is the address range used to deliver interrupts to the FSB. Any device below AIOC, NSI or a PCI Express port may issue a Memory Write to 0FEEx_xxxxh. The IMCH will forward this Memory Write along with its associated data to the FSB as a Message Signaled Interrupt (MSI) transaction. The IMCH terminates the FSB transaction by asserting TRDY# and providing the response. This Memory Write cycle does not go to DRAM.

Reads to this address range are aborted by the IMCH.

10.1.8 High SMM Memory Space

Table 10-10. High SMM Memory Space

	From	To
HIGHSMM	0_FEDA_0000	0_FEDB_FFFF

The HIGHSMM space allows cacheable access to the compatible SMM space by remapping valid SMM accesses between 0_FEDA_0000 and 0_FEDB_FFFF to physical accesses between 0_000A_0000 and 0_000B_FFFF. The accesses are remapped when SMRAM space is enabled, an appropriate access is detected on the processor bus, and when EXSMRC.H_SMRAME (Section 16.1.1.25) allows access to high SMRAM space. Inbound SMM memory accesses from any port are specially terminated; reads are provided with data retrieved from address 0, while writes are ignored entirely (all byte enables deasserted).

10.1.9 PCI Device Memory (MMIO)

The IMCH provides two distinct regions of memory that may be mapped to populated PCI devices. The first is the traditional (non-prefetchable) MMIO range, which must lie below the 4 GByte boundary. The registers associated with non-prefetchable MMIO (MBASE/MLIMIT, see Section 16.4.1.17, “Offset 20h: MBASE - Memory Base Address Register”/Section 16.4.1.18, “Offset 22h: MLIMIT - Memory Limit Address Register”) are unchanged from historical 32-bit architecture IMCH implementations. The second is the prefetchable MMIO range, which has been extended in CMI such that it may lie on either side of the 4 GByte boundary. The registers associated with prefetchable MMIO (PMBASE/PMLIMIT, see Section 16.4.1.19, “Offset 24h: PMBASE - Prefetchable Memory Base Address Register”/Section 16.4.1.20, “Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register”) have been augmented by the PCI defined upper 32-bit base/limit register pair (PMBASU/PMLMTU, see Section 16.4.1.21, “Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register”/Section 16.4.1.22, “Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register”), although only the first nibble of each register is implemented in the IMCH.

The MBASE/MLIMIT pair must be programmed to lie between TOLM and 4 GBytes. The PMBASE/PMLIMIT and PMBASU/PMLMTU registers must be programmed to lie between TOLM and 4 GBytes.

Because these registers define a PCI memory space, they are subject to the memory access enable (MAE) control bit in the standard PCI command register (see Section 16.4.1.4, “Offset 04h: PCICMD - PCI Command Register”).

Note: Using the same address space as both cacheable and non cacheable is discouraged. Also, assigning and writing the same host address space to two independent downstream devices is also discouraged. Although not illegal, both of the above conditions are very difficult to setup intelligently and validate. If 2 devices decide to use the same memory space, and they both send write cycles to it (both either cacheable or uncacheable), there are no guarantees that device 1 data (being older) will get there before device 2 data (being newer) if they do not use a flagging mechanism.

10.1.9.1 Device 2 Memory and Prefetchable Memory

Table 10-11. Device 2 Memory and Prefetchable Memory

	From	To
M2	MBASE2	MLIMIT2
PM2	PMBASE2/PMBASU2	PMLIMIT2/PMLMTU2

Plug-and-play software configures the PEA a memory window in order to provide enough memory space for the devices behind this virtual PCI-to-PCI bridge. Accesses whose addresses fall within these windows are decoded and forwarded to PEA0 for completion. Note that neither region should overlap with any other fixed or relocate-able area of memory. Also note that PCICMD2 refers to PCICMD for device 2.

10.1.9.2 Device 3 Memory and Prefetchable Memory

Table 10-12. Device 3 Memory and Prefetchable Memory

	From	To
M3	MBASE3	MLIMIT3
PM3	PMBASE3/PMBASU3	PMLIMIT3/PMLMTU3

Plug-and-play software configures the PEA1 memory window in order to provide enough memory space for the devices behind this virtual PCI-to-PCI bridge. Accesses whose addresses fall within this window are decoded and forwarded to PEA1 for completion. Note that neither region should overlap with any other fixed or relocate-able area of memory. Also note that PCICMD3 refers to PCICMD for device 3.

Note: If PCI Express Port A0 is configured to operate in x8 mode, all functional space for PEA1 disappears; effectively collapsing M3/PM3 to match the limit addresses of M2/PM2.

10.1.9.3 Device 4 Memory and Prefetchable Memory

Table 10-13. Device 4 Memory and Prefetchable Memory

	From	To
M4	MBASE4	MLIMIT4
PM4	PMBASE4/PMBASU4	PMLIMIT4/PMLMTU4

Plug-and-play software configures the PCI memory window in order to provide enough memory space for the devices behind this virtual PCI-to-PCI bridge. Accesses whose addresses fall within this window are decoded and forwarded to PCI for completion. Note that neither region should overlap with any other fixed or relocate-able area of memory. Also note that PCICMD4 refers to PCICMD for device 4.

10.2 IMCH Responses to EDMA Transactions

In the following tables, the term “Abort” implies that the EDMA engine will immediately stop the transfer in progress. The offending access will not be forwarded to the inbound/outbound arbiter at all, an error bit will be set accordingly, and the error will be escalated as specified by the configuration bits controlling interrupts and errors.

Note: This behavior is quite different from the and PCI Express inbound ports, as in the latter cases the transaction in question was requested by some other initiator elsewhere in the platform. The EDMA engine is a source of traffic all by itself, which makes error containment much simpler in the case of EDMA traffic.

10.2.1 Fixed Address Spaces (EDMA)

Table 10-14 summarizes IMCH responses to EDMA accesses to the various fixed address spaces.

Table 10-14. EDMA Accesses to Fixed Address Spaces

Address Space	Conditions	Destination	IMCH Response
DOSMEM	-	MainMem	Transaction is sent to memory system
VGAA VGAB MDA PAMC0... PAMF0	-	Abort	Programmer’s responsibility not to target EDMA accesses in the legacy region between 640KB and 1 MByte
MEM1_15	-	MainMem	Transaction is sent to memory system
ISA15	FDHC.HEN = 0	MainMem	Hole Disabled: Transaction is sent to memory system
	FDHC.HEN = 1	Abort	Hole Enabled: EDMA will abort on accesses directed to the ISA hole when enabled
MAINMEM	-	MainMem	Transaction is sent to memory system (unless address hits an enabled TSEG SMM range. See TSEGSMM)
TSEGSMM	-	Variable	Refer to **Table 10-16, “Supported SMM Ranges”**.
IOAPIC[0,2-3]	-	Abort	Programmer’s responsibility to avoid the APIC ranges
FSBINTR	-	Abort	Programmer’s responsibility to avoid the FSB interrupt messaging range
HIGHSMM	-	Variable	Refer to **Table 10-16, “Supported SMM Ranges”**.
HIGHMEM	Address is below the top of memory space defined by TOM and the REMAP registers	MainMem	Transaction is sent to memory system.
	Address is above the top of memory	Abort	Hardware will detect an attempt to access above the populated DRAM space, and will abort.

10.2.2 Relocatable Address Spaces (EDMA)

Table 10-15 summarizes IMCH responses to EDMA accesses to the various relocatable address spaces.

Note: EDMA access is not permitted to the port, thus any address mapping to the legacy interface will cause an abort.

Table 10-15. EDMA Accesses to Relocatable Address Spaces

Address Space	Conditions	Destination	IMCH Response
NSI: M NSI: PM	-	Abort	No support for EDMA destination on NSI
PEA: M[n] PEA: PM[n]	Write, MAE = 1	PEA[n]	Transaction forwarded to destination PEA port.
	Write, MAE = 0	Abort	Abort. Memory access disabled.
	Read Transaction	Abort	Abort. No support for peer segment reads.
NSI_SUB	-	Abort	No support for EDMA destination on NSI
PEA: PM[n]	Write, MAE = 1	PEA[n]	Transaction forwarded to destination PEA port.
	Write, MAE = 0	Abort	Abort. Memory access disabled.
	Read Transaction	Abort	Abort. No support for peer segment reads.

10.3 I/O Address Space

The IMCH generates outbound transactions on behalf of all IA-32 core I/O accesses. The IMCH contains two internal registers in the IA-32 core I/O space dedicated to the configuration access mechanism; the Configuration Address Register (CONFIG_ADDRESS) and the Configuration Data Register (CONFIG_DATA). The behavior of the IMCH in response to accesses to these registers is described in Chapter 13.0, “Platform Configuration.”

The IA-32 core allows 64 K+3 bytes to be addressed within the I/O space. The IMCH propagates the IA-32 core I/O address without any translation to the targeted destination bus. Note that the upper three locations can be accessed only during I/O address wrap-around; when signal A16# is asserted on the processor bus. A16# is asserted on the processor bus whenever a Dword I/O access is made from address 0FFFDh, 0FFFEh, or 0FFFFh. In addition, A16# is asserted when software attempts a two-byte I/O access from address 0FFFFh.

All I/O accesses (read or write) which do not map to internal IMCH registers will receive a DEFER response on the FSB, and be forwarded to the appropriate outbound port. The IMCH never posts an I/O write.

The IMCH never responds to inbound transactions to I/O or configuration space initiated on any port. Inbound I/O or configuration transactions requiring a completion are terminated with “master abort” completion packets on the originating port interface. Inbound I/O or configuration write transactions not requiring completion are dropped.

10.3.1 Configuration Window

The I/O addresses 0CF8h and 0CFCh are treated specially, as they define the compatible configuration window. Dword accesses to 0CF8h address the internal IMCH configuration address register. Accesses from 1 to 4 bytes in size to the region from 0CFC-0CFFh are treated as configuration data accesses if configuration space is enabled (bit31 of the configuration address register is set). Refer to Chapter 13.0, “Platform Configuration.” for further details.

10.3.2 VGA and MDA Regions

Along with the memory space address regions described in Section 10.1.2, "VGA and MDA Memory Spaces", there are fixed I/O locations associated with both the VGA and the MDA regions. Accesses to these addresses are routed to NSI by default, but this behavior may be modified via the VGAEN, MDAP and MAE configuration settings. Refer to the prior sections for the rules associated with the configuration settings of VGAEN, MDAP and MAE.

The MDA region includes I/O space addresses 3B4h, 3B5h, 3B8h, 3B9h, 3BAh and 3BFh. The VGA region includes I/O space ranges 3B0-3BBh, and 3C0-3DFh. The *PCI Specification* defines both MDA and VGA to be 10-bit address decode, thus all accesses with A[9:0] matching any of these addresses are subject to the associated routing rules. Address bits A[15:10] are ignored when the check against these fixed addresses are applied.

The order of precedence for the routing checks is as follows:

- MAE = 0, MDA addresses will route to NSIif MDAP is set, overriding any VGAEN.
- MAE = 1, MDA addresses will route to the Peer device if MDAP is set, overriding any VGAEN.
- MAE = 0, VGA addresses will route to the NSI.
- MAE = 1, VGA addresses will route to the PCI Express port with its VGAEN set, if any.
- MAE = 0, MDA addresses which fall within VGA regions will route to the NSI if MDAP is clear.
- MAE = 1, MDA addresses which fall within VGA regions will follow VGAEN if MDAP is clear.
- Both VGA and MDA addresses default to NSI if MDAP and all VGAEN bits are clear.

Note: Setting of MDAP or any of the VGAEN bits implies that the ISAEN bit is also set in all virtual P2P bridges, because of the 10-bit decode requirement.

Note: AIOC access to this space is not supported. Upstream attempted accesses to this space are programming errors and will result in a master abort.

10.4 Main Memory Addressing

The “High Memory” and “Extended Memory” address regions are together called “Main Memory.” Main memory is composed of address segments that refer to DDR SDRAM system memory. Main memory addresses are mapped to DDR SDRAM channels, devices, banks, rows, and columns in different ways depending upon the type of memory being used and upon the density or organization of the memory. The process for determining the device and channel IDs for addressed devices is as follows:

- The requested address is compared against the values of all eight DRB registers. The number of the register whose programmed value is greater than the address and whose previous register is less than the address is the output of the comparison.
- The value of the DRB register “below” is subtracted from the address in order to determine the offset into the group.
- The offset determines the manner in which the row, column, and bank address bits are extracted from the address.

10.5 System Management Mode (SMM) Space

CMI supports the use of main memory as System Management RAM (SMM RAM) enabling the use of System Management Mode. The IMCH supports three SMM options:

- Compatible SMRAM (C_SMRAM)
- High Segment (HSEG)
- Top of Memory Segment (TSEG)

System Management RAM space provides an access protected memory area that is available for SMI handler code and data storage. This memory resource is normally hidden from the Operating System so that the processor has immediate access to this memory space upon entry to SMM (cannot be swapped-out).

10.5.1 SMM Addressing Ranges

IMCH provides three SMRAM options:

- Below 1 MByte option that supports compatible SMI handlers.
- Above 1 MByte option that allows new SMI handlers to execute with write-back cacheable SMRAM.
- Optional larger write-back cacheable T_SEG area from 128 Kbyte to 1 MByte in size. The above 1 MByte solutions require changes to compatible SMRAM handler code to properly execute above 1 MByte.

Note: The first two options both map legal accesses to the same physical range of memory, while the third defines an independent region of addresses.

10.5.1.1 SMM Space Restrictions

If any of the following conditions are violated, the results of SMM accesses are unpredictable and may cause the system to hang:

- The Compatible SMM space must not be set-up as cacheable.
- Both D_OPEN and D_CLOSE must not be set to 1 at the same time.
- When TSEG SMM space is enabled, the TSEG space must not be reported to the OS as available DRAM. This is a BIOS responsibility.

BIOS and SMM code must cooperate to properly configure the IMCH in order to ensure reliable operation of the SMM function.

10.5.1.2 SMM Space Definition

SMM space is defined by both its addressed SMM space and its DRAM SMM space. The addressed SMM space is defined as the range of FSB addresses used by the IA-32 core to access SMM space. DRAM SMM space is defined as the range of physical DRAM memory locations containing SMM information.

The SMM space can be accessed at one of three transaction address ranges:

- Compatible
- High
- TSEG

The Compatible and TSEG SMM space is not remapped and therefore the addressed and DRAM SMM physical addresses are identical. The High SMM space is remapped; thus the addressed and DRAM SMM locations are different. Note that the High DRAM space is the same as the Compatible Transaction Address space.

Table 10-16 describes all three unique addressing combinations:

- Compatible Transaction Address
- High Transaction Address
- TSEG Transaction Address

Table 10-16. Supported SMM Ranges

SMM Space Enabled	Transaction Address Space (Adr)	DRAM Space (DRAM)
Compatible (C)	A0000h to BFFFFh	A0000h to BFFFFh
High (H)	0FEDA0000h to 0FEDBFFFFh	A0000h to BFFFFh
TSEG (T)	(TOLM-TSEG_SZ) to TOLM	(TOLM-TSEG_SZ) to TOLM

Notes:

1. High SMM: This implementation is consistent with the Intel E7500 and Intel E7501 designs. In prior MCH designs the High segment was the 384 Kbyte region from A_0000h to F_FFFFh. However C_0000h to F_FFFFh was not useful, so it has been deleted in the IMCH design.
2. TSEG SMM: This implementation is consistent with the Intel E7500 and Intel E7501 designs. In prior MCH designs the TSEG address space was offset by 256 MBytes to allow for simpler decoding and the TSEG was remapped to just under the TOLM. In the IMCH the TSEG region is not offset by 256 MBytes and it is not remapped.
3. In Cases where DRAM TOLM is less than TOM TSEG cannot be used for SMM. For this case MENC memory spans consecutive space from above TOLM to below TOLM and will conflict with TSEG space

10.6 Memory Reclaim Background

The following Memory Mapped I/O devices and ranges are typically located below 4 GBytes:

- High BIOS
- H-Seg
- XAPIC
- Local APIC
- FSB Interrupts
- PEA0 through PEA1 M, PM and BAR regions

In previous generation MCH architectures, the physical DRAM memory overlapped by the logical address space allocated to these Memory Mapped I/O devices was unusable. In server systems the memory allocated to memory mapped I/O devices could easily exceed 1 GByte. This creates the possibility of a large amount of physical memory populated in the system becoming unusable.

The IMCH provides the capability to reclaim the physical memory overlapped by the Memory Mapped I/O logical address space via remapping physical memory from the Top of Low Memory (TOLM) boundary up the 4 GBytes boundary (or TOM if less than 4 GBytes) to an equivalent sized logical address range located just above the top of physical memory

10.6.1 Memory Remapping Algorithm

Note: The IA-32 core is not capable of using the remap capability and care must be taken to ensure that no addresses sent to the IA-32 core are greater than TOLM in that it will cause aliasing. The remap capability can be utilized by the AIOC, IICH and PCIe ports.

Terminology clarification:

Physical Address	The address presented to the IMCH is traditionally called a “physical address,” because Intel architecture processors contain both segmentation and paging hardware, and all compatible software differentiates between logical addresses, virtual addresses, and physical addresses. The algorithm for remapping addresses presented to the IMCH to reclaim DRAM address space must be implemented such that the mechanism is invisible to compatible software.
System Address	The system address applies to the internal IMCH interface to physical DRAM memory, and is not directly visible to software, other than through certain internal logging registers used to store decoded DRAM address information for error isolation.

An incoming address (referred to as a physical address) is checked to see if it falls in the memory remap window. The bottom of the remap window is defined by the value in the REMAPBASE register (see Section 16.1.1.31, “Offset C6h: REMAPBASE - Remap Base Address Register”). The top of the remap window is defined by the value in the REMAPLIMIT register (Section 16.1.1.32, “Offset C8h: REMAPLIMIT – Remap Limit Address Register”). An address that falls within this window is remapped to the physical memory starting at the address defined by the TOLM register.

10.7 IICH Register and Memory Mappings

This section covers CMI's IICH various address decoding ranges.

This section is for background purposes, and must not to be considered by implementers and validators as part of the behavioral definition of CMI. Each decode range is described elsewhere in the section associated with the corresponding function.

10.7.1 I/O Map

The I/O map is divided into separate types. Fixed ranges cannot be moved, but in some cases can be disabled. Variable ranges can be moved and can also be disabled.

10.7.1.1 Fixed I/O Address Ranges

Table 10-17 shows the Fixed I/O decode ranges from the IA-32 core perspective. Note that for each I/O range, there may be separate behavior for reads and writes. Cycles that go to target ranges that are marked as Reserved will not be decoded, and are passed to PCI to PCI Bridge where they are dropped.

Address ranges that are not listed or marked reserved are NOT positively decoded by the IICH (unless assigned to one of the variable ranges). In subtractive mode, I/O ranges that are not otherwise decoded are forwarded to PCI to PCI Bridge where they are dropped.

Table 10-17. Fixed I/O Ranges Decoded by IICH (Sheet 1 of 3)

I/O Address	Read Target	Write Target	Internal Unit	Separate Enable/ Disable
00h – 08h	DMA Controller	DMA Controller	DMA	None
09h – 0Eh	RESERVED	DMA Controller	DMA	None
0Fh	DMA Controller	DMA Controller	DMA	None
10h – 18h	DMA Controller	DMA Controller	DMA	None
19h – 1Eh	RESERVED	DMA Controller	DMA	None
1Fh	DMA Controller	DMA Controller	DMA	None
20h – 21h	Interrupt Controller	Interrupt Controller	Interrupt	None
24h – 25h	Interrupt Controller	Interrupt Controller	Interrupt	None
28h – 29h	Interrupt Controller	Interrupt Controller	Interrupt	None
2Ch – 2Dh	Interrupt Controller	Interrupt Controller	Interrupt	None
2E – 2F	LPC SIO	LPC SIO	Forwarded to LPC	Yes
30h – 31h	Interrupt Controller	Interrupt Controller	Interrupt	None
34h – 35h	Interrupt Controller	Interrupt Controller	Interrupt	None
38h – 39h	Interrupt Controller	Interrupt Controller	Interrupt	None
3Ch – 3Dh	Interrupt Controller	Interrupt Controller	Interrupt	None
40h – 42h	Timer/Counter	Timer/Counter	PIT (8254)	None
43h	RESERVED	Timer/Counter	PIT	None
4E – 4F	LPC SIO	LPC SIO	Forwarded to LPC	Yes
50h – 52h	Timer/Counter	Timer/Counter	PIT	None
53h	RESERVED	Timer/Counter	PIT	None
60h	Microcontroller	Microcontroller	Forwarded to LPC	Yes w/ 64h

Table 10-17. Fixed I/O Ranges Decoded by IICH (Sheet 2 of 3)

I/O Address	Read Target	Write Target	Internal Unit	Separate Enable/ Disable
61h	NMI Controller	NMI Controller	IA-32 core Interface	None
62h	Microcontroller	Microcontroller	Forwarded to LPC	Yes w/ 66h
63h	NMI Controller[1]	NMI Controller[1]	IA-32 core Interface	Yes, alias to 61h
64h	Micocontroller	Microcontroller	Forwarded to LPC	Yes w/ 60h
65h	NMI Controller[1]	NMI Controller[1]	IA-32 core Interface	Yes, alias to 61h
66h	Microcontroller	Microcontroller	Forwarded to LPC	Yes w/ 62h
67h	NMI Controller[1]	NMI Controller[1]	IA-32 core Interface	Yes, alias to 61h
70h	RESERVED	NMI and RTC Controller	RTC	None
71h	RTC Controller	RTC Controller	RTC	None
72h	RTC Controller	NMI and RTC Controller	RTC	Yes, w/ 73h
73h	RTC Controller	RTC Controller	RTC	Yes, w/ 72h
74h	RTC Controller	NMI and RTC Controller	RTC	None
75h	RTC Controller	RTC Controller	RTC	None
76h	RTC Controller	NMI and RTC Controller	RTC	None
77h	RTC Controller	RTC Controller	RTC	None
80h	DMA Controller, or LPC	DMA Controller, or LPC	DMA	None
81h – 83h	DMA Controller	DMA Controller	DMA	None
84h – 86h	DMA Controller	DMA Controller and LPC	DMA	None
87h	DMA Controller	DMA Controller	DMA	None
88h	DMA Controller	DMA Controller and LPC	DMA	None
89h – 8Bh	DMA Controller	DMA Controller	DMA	None
8Ch – 8Eh	DMA Controller	DMA Controller and LPC	DMA	None
8Fh	DMA Controller	DMA Controller	DMA	None
90h – 91h	DMA Controller	DMA Controller	DMA	Yes, alias to 8xh
92h	Reset Generator	Reset Generator	IA-32 core Interface	None
93h – 9Fh	DMA Controller	DMA Controller	DMA	Yes, alias to 8xh
A0h – A1h	Interrupt Controller	Interrupt Controller	Interrupt	None
A4h – A5h	Interrupt Controller	Interrupt Controller	Interrupt	None
A8h – A9h	Interrupt Controller	Interrupt Controller	Interrupt	None
ACh – ADh	Interrupt Controller	Interrupt Controller	Interrupt	None
B0h – B1h	Interrupt Controller	Interrupt Controller	Interrupt	None
B2h – B3h	Power Management	Power Management	Power Management	None
B4h – B5h	Interrupt Controller	Interrupt Controller	Interrupt	None
B8h - B9h	Interrupt Controller	Interrupt Controller	Interrupt	None
BCh – BDh	Interrupt Controller	Interrupt Controller	Interrupt	None
C0h – D1h	DMA Controller	DMA Controller	DMA	None
D2h – DDh	RESERVED	DMA Controller	DMA	None
DEh – DFh	DMA Controller	DMA Controller	DMA	None
170h – 177h	SATA	SATA	SATA	Yes

Table 10-17. Fixed I/O Ranges Decoded by IICH (Sheet 3 of 3)

I/O Address	Read Target	Write Target	Internal Unit	Separate Enable/ Disable
1F0h – 1F7h	SATA	SATA	SATA	Yes
200 – 207h	Gameport Low	Gameport Low	Forwarded to LPC	Yes
208 – 20Fh	Gameport High	Gameport High	Forwarded to LPC	Yes
376h	SATA	SATA	SATA	Yes
3F6h	SATA	SATA	SATA	Yes
4D0h – 4D1h	Interrupt Controller	Interrupt Controller	Interrupt	None
CF9h	Reset Generator	Reset Generator	IA-32 core Interface	None

Notes:
1. Only if the Port 61 Alias Enable bit (GCS.P61AE) bit is set. Otherwise, the target is dropped.
2.

10.7.1.2 Variable I/O Decode Ranges

Table 10-18 shows the Variable I/O Decode Ranges. They are set using Base Address Registers (BARs) or other configuration bits in the various configuration spaces. The PnP software (PCI or ACPI) can use their configuration mechanisms to set and adjust these values.

Warning: The Variable I/O Ranges must not be set to conflict with the Fixed I/O Ranges. If the configuration software allows conflicts to occur, it may produce unpredictable results. There are no checks for conflicts.

Table 10-18. Variable I/O Decode Ranges

Range Name	Mappable	Size (Bytes)	Target
ACPI	Anywhere in 64K I/O Space	64	Power Management
USB #1	Anywhere in 64K I/O Space (See Note 1)	32	USB1 Host Controller 1
SMBus	Anywhere in 64K I/O Space	32	SMB Unit
TCO	96 bytes above ACPI base	32	TCO Unit
GPIO	Anywhere in 64K I/O space	64	GPIO Unit
Parallel Port	3 ranges in 64K I/O Space	8[2]	LPC Peripheral
Serial Port 1	8 Ranges in 64K I/O Space	8	LPC Peripheral
Serial Port 2	8 Ranges in 64K I/O Space	8	LPC Peripheral
Floppy Disk Controller	2 Ranges in 64K I/O Space	8	LPC Peripheral
LPC Generic 1	Anywhere in 64K I/O Space	128	LPC Peripheral
LPC Generic 2	Anywhere in 64K I/O Space	16	LPC Peripheral
I/O Trapping Ranges	Anywhere in 64K I/O Space	1 to 256 Bytes	Trap on Internal I/O Data Bus

Notes:
1. These ranges are decoded directly.
2. There is also an alias 400h above the parallel port range that is used for ECP parallel ports.

10.7.2 Memory Map

Table 10-19 shows (from the IA-32 core perspective) the memory ranges that are decoded. Cycles that arrive that are not directed to any of the internal memory targets that decode (see Table 10-19)are dropped.

Software must not attempt locks to the IICH's memory-mapped I/O ranges for USB 2.0, and HPET (High Precision Event Timer). If attempted, the lock is not honored which means potential deadlock conditions may occur.

Table 10-19. IICH Memory Decode Ranges (from IA-32 core Perspective)

Memory Range	Target	Dependency/Comments
000E0000 - 000EFFFF	FWH	Bit 6 in FWH Decode Enable Register is set.
000F0000 - 000FFFFF	FWH	Bit 7 in FWH Decode Enable Register is set
FEC00000 - FEC0 0040	I/O(x)APIC inside IICH	
FFC0 0000 - FFC7 FFFF FF80 0000 - FF87 FFFF	FWH	Bit 8 in FWH Decode Enable Register
FFC8 0000 – FFCF FFFF FF88 0000 - FF8F FFFF	FWH	Bit 9 in FWH Decode Enable Register
FFD0 0000 - FFD7 FFFF FF90 0000 - FF97 FFFF	FWH	Bit 10 in FWH Decode Enable Register is set
FFD8 0000 – FFDF FFFF FF98 0000 - FF9F FFFF	FWH	Bit 11 in FWH Decode Enable Register is set
FFE0 000 - FFE7 FFFF FFA0 0000 - FFA7 FFFF	FWH	Bit 12 in FWH Decode Enable Register is set
FFE8 0000 – FFEF FFFF FFA8 0000 – FFAF FFFF	FWH	Bit 13 in FWH Decode Enable Register is set
FFF0 0000 - FFF7 FFFF FFB0 0000 - FFB7 FFFF	FWH	Bit 14 in FWH Decode Enable Register is set
FFF8 0000 – FFFF FFFF FFB8 0000 – FFBF FFFF	FWH	Always enabled. The top two 64KB blocks in this range can be swapped by the IICH. See Section 10.5 for details.
FF70 0000 - FF7F FFFF FF30 0000 - FF3F FFFF	FWH	Bit 3 in FWH Decode Enable 2 Register is set
FF60 0000 - FF6F FFFF FF20 0000 - FF2F FFFF	FWH	Bit 2 in FWH Decode Enable 2 Register is set
FF50 0000 - FF5F FFFF FF10 0000 - FF1F FFFF	FWH	Bit 1 in FWH Decode Enable 2 Register is set
FF40 0000 - FF4F FFFF FF00 0000 - FF0F FFFF	FWH	Bit 0 in FWH Decode Enable 2 Register is set
1KB anywhere in 4GB range	USB 2.0 Host Controller	Enable via standard PCI mechanism (Device 29, Function 7)
FED0 X000h-FED0 X3FFh	HPET	BIOS determines "fixed" location which is one of four 1KB ranges where X (in the first column) is 0h, 1h, 2h, or 3h.
All other	N/A	Master aborted

10.7.3 Boot-Block Update Scheme

The IICH supports a “Top-Block Swap” mode that swaps the top block in the FWH (the boot block) with another location. This allows for safe update of the Boot Block (even if a power failure occurs). When the “top-swap” enable bit is set, inverts A16 for cycles going to the upper two 64 Kbyte blocks in the FWH. Specifically, in this mode, accesses to FFFF_0000h-FFFF_FFFFh are directed to FFFE_0000h-FFFE_FFFFh and vice versa. When the Top Swap Enable bit is 0, the IICH will not invert A16. This bit is automatically set to 0 by RTEST#, but not by PLTRST#.

The scheme is based on the concept that the top block is reserved as the “boot” block, and the block immediately below the top block is reserved for doing boot-block updates.

The algorithm is:

1. Software copies the top block to the block immediately below the top.
2. Software checks that the copied block is correct. This could be done by performing a checksum calculation.
3. Software sets the “Top-Block Swap” bit. This will invert A16 for cycles going to the FWH.
4. Software erases the top block.
5. Software writes the new top block.
6. Software checks the new top block.
7. Software clears the top-block swap bit.
8. Software sets the Top_Swap Lock-Down bit.

If a power failure occurs at any point after step 3, the system is able to boot from the copy of the boot block that is stored in the block below the top. This is because the top-swap bit is backed in the RTC well.

Note: The top-block swap mode may be forced by an external strapping option (see Chapter 16.0, “IMCH Registers.”). When top-block swap mode is forced in this manner, the TOP_SWAP bit cannot be cleared by software. A reboot with the strap removed will be required to exit a forced top-block weap mode.

Note: Top-block swap mode only affects accesses to the Firmware Hub space, not feature space.

Note: The top-block swap mode has no effect on accesses below FFFE_0000h.

§ §

11.0 System Memory Controller

11.1 Overview

The memory controller is responsible for controlling the off-chip DRAM devices. The unit scheduler, control and protocol state machines access DRAM devices over a wide range of speed bins using DDR2 DRAM technology. The EP80579 supports one memory channel.

11.2 Memory Controller Feature List

The memory controller supports the following features

- Supports 1 DIMM
  - — DDR2
  - — 64 and 32-bit mode.
  - — Single rank (64 or 32-bit mode) or Dual rank (64 bit mode) device.
  - — DDR2-400, DDR2-533, DDR2-667, DDR2-800.
  - — See to “Rules for Populating DIMM Slots” for more details.
- Supports dual DIMMs
  - — DDR2 and 64 bit mode only
  - — Registered Dual DIMM support with 1T command/address timing. Unbuffered Dual DIMM support with 2T command/address timing. Please refer to “2T Timing Mode” for details.
  - — 32-bit mode dual DIMM is not supported
  - — Restrictions on speed, number of ranks and modes in 2 DIMM mode, refer to “Rules for Populating DIMM Slots” for details.
  - — DDR2-400 is supported.
  - — DDR2-533 and DDR2-667 are also supported with special design guidelines. Refer to Table 11-5 for more details.
- Supports 32-bit mode
  - — 32-bit mode is for memory-down configuration, thus the following modes are not supported: dual DIMM, dual rank, registered
- Supports 256 Mb, 512 Mb, 1 Gb and 2 Gb density parts in the x8 configuration. Table 11-1 shows the various DDR2 device densities and widths supported. Table 11-2 and Table 11-3 shows the various memory capacity configurations supported using these parts in the 64 bit and 32-bit modes.
  - — Memory Controller does not support 4 Gb density
  - — See Section 11.3, “Configurations” for more details.
- Supports 4-bank devices
  - — 256 Mb and 512 Mb DDR2 parts

- Supports 8-bank devices.
  - 1 Gb and 2 Gb DDR2 parts
- Supports unbuffered and registered DIMMs
- Supports discrete memory components soldered on the board
  - supports 2 DQ loads similar to the double sided DIMM configuration.
- One 72b wide DRAM interface, 64b data + 8b ECC.
  - The 64b data interface can optionally configured to be a 32b interface with 8b ECC. The 32b option provides half the total bandwidth compared to the 64b interface, and meets the requirements for systems that are cost sensitive and want to populate small memory footprint soldered on the mother board.
- Supports burst-of-4 mode with a minimum access size of 32B for the 64b interface.
  - On the 64b interface accesses longer than 32B are serviced as multiple burst-of-4 transactions without closing the page.
  - On the 32b interface, the controller supports burst-of-8 for a minimum access size that is still 32B.
- Optional error protection using ECC bits and error code that supports SEC/DED (single bit error correction/double bit error detection). Please see "Offset 7Ch: DRC – DRAM Controller Mode Register" for details.
  - On a single bit error, the memory controller corrects the error bit and writes back the correct data value to DRAM, if enabled by software selection.
  - Does not support DED (double-bit error detect) retries.
- Supports maximum of 4 GB DRAM capacity as shown in Table 11-2.
  - One memory channel
- Support for demand scrub in hardware
  - Refer to Section 16.1.1.45, "Offset 88h: SDRC – DDR SDRAM Secondary Control Register" on page 439 for details.
- Support for background scrubbing in hardware.
  - Programmable hardware scrub engine that allows background scrub at a wide range of rates, including, but not limited to: up to 4 GB every hour, day, week, or very fast rates, mainly used for validation purposes.

The memory controller supports the following transactions

- Simple read transactions.
  - Supported lengths - 1-7B, 8B, 16B, 24B, 32B, 64B
  - Read transactions smaller than 32B will result in a full 32B read on the interface
- Simple write transactions.
  - Supported lengths - 8B, 16B, 24B, 32B, 64B
  - Writes of lengths 8B, 16B and 24B will take the same time on the interface as a 32B write, the actual bytes that are written are specified by the DQ masks
  - Writes of length 40B, 48B, 56B will be treated as a 32B write followed by an 8B, 16B or 24B write
  - Writes that are smaller than 8B (i.e 1-7B) require a read-modify-write operation and these will be supported upstream in the pipeline in other units. The memory controller will not support read-modify-write operation.
- CSR reads and writes

- The memory controller does not support any atomic operations. Other units upstream support atomic operations.
- All memory controller operations are aligned on a 64B boundary
- The memory controller interleaves the memory across banks on a 64B boundary
  - — Any transaction that crosses a 64B alignment boundary or is larger than 64B must be split by the upstream agent.
- Supports external DIMMs or SO-DIMMs.

11.3 Configurations

Table 11-1 shows the various DDR2 device densities and widths supported by the memory controller.

Table 11-1. Supported DDR2 Device Densities and Width

Density (Mb)	DDR2x8
256	Supported
512	Supported
1024	Supported
2048	Supported
4096	Not Supported

Table 11-2 shows the various capacity configurations supported in 64b mode. The first column shows the total DRAM capacity on the channel. The rest of the columns indicate the DRAM devices features, densities and the number of devices required to achieve the given capacity. A single sided DIMM is indicated by no parts populated on side B. Double sided DIMM has parts populated on both sides.

For each configuration, an additional DRAM part per side is required to support ECC bits. A x8 part provides all the bits required for ECC. Note that memory system can be built without ECC enabled.

In the 64b configuration, the minimum capacity supported is 256 MB and the maximum capacity supported is 4 GB.

Table 11-2. Supported DRAM Capacity for 64b Mode

Total DRAM Capacity	DRAM Density	DRAM Part Width	Total # of parts on side A (w/o ECC)	Total # of parts on side B (w/o ECC)
256 MB	256 Mb	x8	8	0
512 MB	256 Mb	x8	8	8
	512 Mb	x8	8	0
1 GB	512 Mb	x8	8	8
	1 Gb	x8	8	0
2 GB	1 Gb	x8	8	8
	2 Gb	x8	8	0
4 GB	2 Gb	x8	8	8

Table 11-3 shows the various configurations supported in the 32b mode.In the 32-bit mode only single rank DDR2 devices are supported. In this mode, the minimum capacity supported is 128 MB and the maximum capacity supported is 1 GB.

Table 11-3. Supported DRAM Capacity for 32b Mode

Total DRAM Capacity	DRAM Density	DRAM Part Width	Total # of parts on side A (w/o ECC)	Total # of parts on side B (w/o ECC)
128 MB	256 Mb	x8	4	0
256M B	512 Mb	x8	4	0
512 MB	1 Gb	x8	4	0
1 GB	2 Gb	x8	4	0

Note: The EP80579 supports only single ranks in 32-bit mode.

Table 11-4 shows the supported DIMM raw cards in the registered and unbuffered formats.

Table 11-4. Raw Cards Supported by the EP80579

Raw Card	Number of DDR2 SDRAMs	SDRAM Organization	Number of Ranks	
A	9	x8, planar, single row	1	RDIMM
F	9		1	
B	18		2	
G	18		2	
D	8		1	UDIMM
F	9		1	
E	16		2	
G	18		2	

Table 11-5 shows the supported DDR2device speed grades for single and Dual DIMMs. Dual DIMM support uses 2N or 2T command/address timing.

Table 11-5. Supported DDR2 Data Speeds

DDR Speed	1 DIMM 1 rank	1 DIMM 2 ranks	2 DIMMs 1 rank each
DDR2-400	R = 1T UB = 1T	R = 1T UB = 1T	R = 1T UB = 2T
DDR2-533	R = 1T UB = 1T	R = 1T UB = 1T	R = 1T UB = 2T
DDR2-667	R = 1T UB = 1T	R = 1T UB = 1T	R = 1T UB = 2T
DDR2-800	R = 1T UB = 2T	R = 1T UB = 2T	Not supported
R = Registered UB = Unbuffered 1T = 1T Address/Command Timing 2T = 2T Address/Command Timing			

11.3.1 Rules for Populating DIMM Slots

1. In all configurations, the speed and timing will be the lowest among the 2 DIMMs, as determined by the SPD registers on the DIMMs
2. Dual DIMM mode supports a subset of the DDR2 data speeds as shown in Table 11-5.
3. Table 11-6 shows the supported DIMM population of the 2 ranks. All of configurations not shown in Table 11-6 are not supported.The rank configurations supported are one or two ranks on a single DIMM or one rank on each of the 2 DIMMs. Two ranks in each of the 2 DIMMs (i.e., 4 ranks) is not supported.

Table 11-6. Supported DIMM Populations

	DIMM 1		DIMM 0	
	Rank 0	Rank 1	Rank 0	Rank 1
1 - Single Rank	Empty	Empty	DRB0 CS0 ODT0	Empty
1 - Dual Rank	Empty	Empty	DRB0 CS0 ODT0	DRB2 CS1 ODT1
2 - Single Rank	DRB2 CS1 ODT1	Empty	DRB0 CS0 ODT0	Empty

TABLE KEY:
ODT0/ODT1: ODT pins. Please refer to Section 11.4.2 for more information.
CS0/CS1: Chip Selects
DRB: Dram Row Boundary Register

4. Table 11-7 shows the supported rank configurations when using 2 ranks for the single and dual DIMM.

Table 11-7. Supported Rank Configurations in Single and Dual DIMM mode

Single DIMM (DDR2) (64 bits - rank 0 & rank 1) (32 bits - rank 0 only)		Dual DIMM (DDR2, 64 bit only)	
Rank 0	Rank 1	Rank 0, DIMM 0	Rank 0, DIMM 1
128 MB (32 bit only)	NA	NA	NA
256 MB	256 MB	256 MB	256 MB, 512 MB, 1 GB, 2 GB
512 MB	512 MB	512 MB	256 MB, 512 MB, 1 GB, 2 GB
1 GB	1 GB	1 GB	256 MB, 512 MB, 1 GB, 2 GB
2 GB	2 GB	2 GB	256 MB, 512 MB, 1 GB, 2 GB

5. There are additional restrictions when 2 ranks are used. Both the ranks need to be:
   — Either registered or unbuffered. No mixing of registered and unbuffered ranks.
   — 64 bit mode only. 32-bit mode is not supported in dual rank.
6. For 32-bit mode, only discrete components of single rank are supported. Two ranks in 32-bit mode is not supported. Table 11-3 shows the supported configurations in the 32-bit mode.

11.3.2 DRAM Addressing

Table 11-8, Table 11-9, Table 11-10 and Table 11-11 show the DRAM device addressing for the various DDR2 device densities supported by the memory controller.

Note that:

- x4 and x 16 devices are not supported.
- 4Gb and higher device density parts are not supported.
- See Table 11-7 for the supported device densities and widths.

Table 11-8. 256Mb Addressing

Configuration	DDR2 32 Mb x 8
# of Banks	4
Bank Address	BA0, BA1
Auto Precharge	A10
Row Address	A0-A12
Column Address	A0-A9
Page Size	1KB

Table 11-9. 512Mb Addressing

Configuration	DDR2 64 Mb x 8
# of Banks	4
Bank Address	BA0, BA1
Auto Precharge	A10
Row Address	A0-A13
Column Address	A0-A9
Page Size	1KB

Table 11-10. 1Gb Addressing

Configuration	DDR2 128 Mb x 8
# of Banks	8
Bank Address	BA0-BA2
Auto Precharge	A10

Table 11-10. 1Gb Addressing

Configuration	DDR2 128 Mb x 8
Row Address	A0-A13
Column Address	A0-A9
Page Size	1KB

Table 11-11. 2Gb Addressing

Configuration	DDR2 256 Mb x 8
# of Banks	8
Bank Address	BA0-BA2
Auto Precharge	A10
Row Address	A0-A14
Column Address	A0-A9
Page Size	1KB

11.3.3 Memory Address Translation Tables

Section 11.3.3.1 shows the address bit translation from the system address to the DRAM row/column/bank address for the different DDR2 configurations supported by the memory controller.

256 Mb, 512 Mb, 1024 Mb and 2048 Mb DRAM device densities are shown but please refer to Table 11-1 for supported DDR2 device densities and widths. The memory capacity that can be achieved for each device density in the single and dual rank mode is also shown in the address mapping tables.

11.3.3.1 DDR2 Address Translation Tables

Figure 11-1 shows the translation tables for 64 bit, burst size 4 devices in x8 width. Note: only burst 4 is supported for a DDR2 64 data bit interface.

Figure 11-2 shows the translation tables for 32 bit, burst size 8 devices in x8 width. Note: only burst 8 is supported for a DDR2 32 data bit interface.

For a list of memory controller supported DDR2 device types and widths see Table 11-1.

Figure 11-1. Memory Address Tables for 64 Bit, Burst Size 4 and x8 DDR2 Devices

Total Cap (MB) Single Rank	Device density (Mb)	Device width	Bus Width	R/C/B addr lines		BA2	BA1	BA0	A14	A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
256	256	8	64	13x10x2	Row	0	7	6	0	0	26	25	24	23	22	21	20	19	18	17	16	15	14
					Col				0	0	0	0	AP	27	13	12	11	10	9	8	5	0	0
512	512	8	64	14x10x2	Row	0	7	6	0	28	26	25	24	23	22	21	20	19	18	17	16	15	14
					Col				0	0	0	0	AP	27	13	12	11	10	9	8	5	0	0
1024	1024	8	64	14x10x3	Row	8	7	6	0	28	26	25	24	23	22	21	20	19	18	17	16	15	14
					Col				0	0	0	0	AP	27	13	12	11	10	9	29	5	0	0
2048	2048	8	64	15x10x3	Row	8	7	6	30	28	26	25	24	23	22	21	20	19	18	17	16	15	14
					Col				0	0	0	0	AP	27	13	12	11	10	9	29	5	0	0

Figure 11-2. Memory Address Tables for 32 Bit, Burst Size 8 and x8 DDR2 Devices

Device density (Mb)	Device width	Bus Width	R/C/B addr lines		BA2	BA1	BA0	A14	A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
256	8	32	13x10x2	Row	0	7	6	0	0	26	25	24	23	22	21	20	19	18	17	16	15	14
				Col				0	0	0	0	AP	13	5	12	11	10	9	8	0	0	0
512	8	32	14x10x2	Row	0	7	6	0	27	26	25	24	23	22	21	20	19	18	17	16	15	14
				Col				0	0	0	0	AP	13	5	12	11	10	9	8	0	0	0
1024	8	32	14x10x3	Row	8	7	6	0	27	26	25	24	23	22	21	20	19	18	17	16	15	14
				Col				0	0	0	0	AP	13	5	12	11	10	9	28	0	0	0
2048	8	32	15x10x3	Row	8	7	6	29	27	26	25	24	23	22	21	20	19	18	17	16	15	14
				Col				0	0	0	0	AP	13	5	12	11	10	9	28	0	0	0

11.3.4 DRAM Timings

The EP80579 is highly configurable in its DRAM timing configuration, but only a limited subset of the setting combinations possible are verified by Intel. The approved and expected settings for the various flavors of supported memory are listed in Table 11-12. For more information on the DRT register see "Offset 78h: DRT0 - DRAM Timing Register 0" and "Offset 64h: DRT1 – DRAM Timing Register 1".

Table 11-12. Supported DRAM Timings

Memory Speed	CL (CAS latency)	tRCD (RAS-CAS delay)	tRP (RAS Precharge)
DDR2-400[a]	3	3	3
DDR2-400[b]	4	4	4
DDR2-533	4	4	4
DDR2-667	5	5	5
DDR2-800	5	5	5
	6	6	6

a: 3-3-3 is not supported for systems which require ODT
b: May be accomplished by programming 3-3-3 parts to 4-4-4

11.3.4.1 2T Timing Mode

The address and command group (Bank Address BA[2:0], Address MA[14:0], Command pins RAS_L, CAS_L and WE_L) of pins to the DRAM devices can be clocked either by 1T or 2T timing as shown in Figure 11-3. The control group of pins (CKE, CS, ODT) are not impacted by the 2T timing mode.

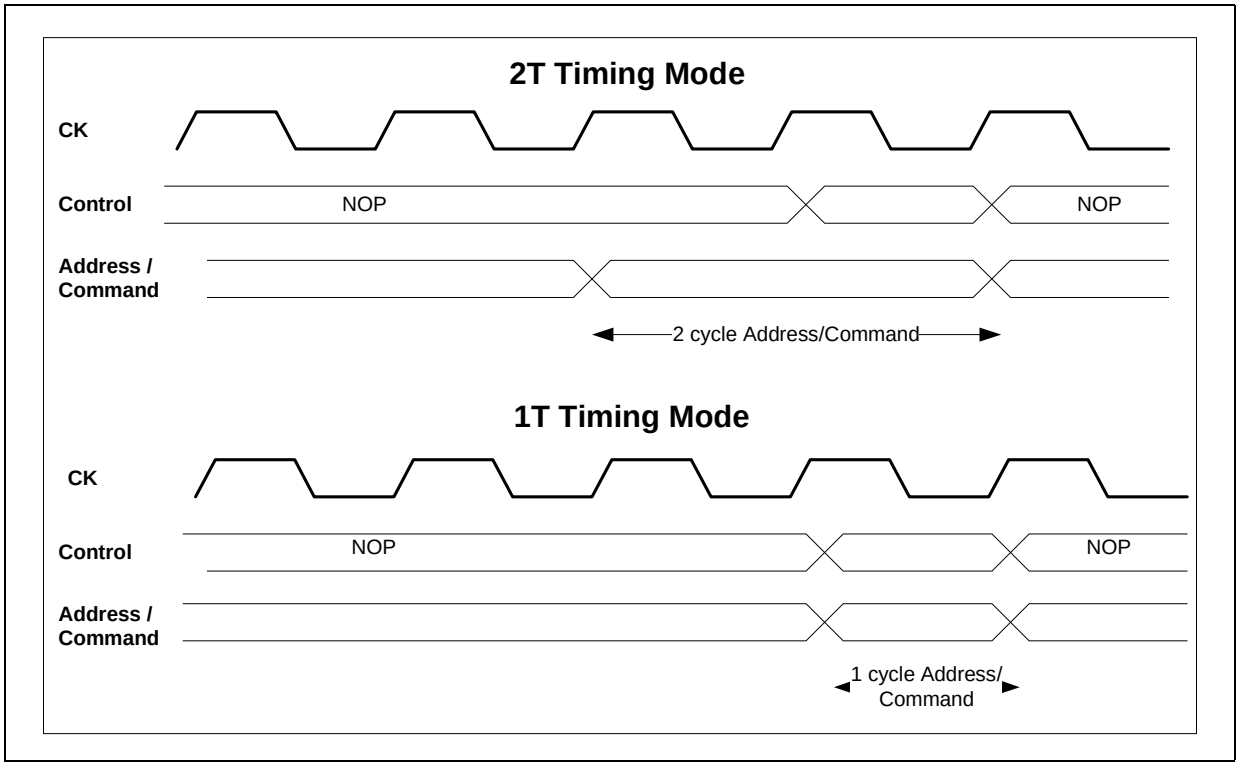
When in 1T timing mode, a new command can be issued to the DRAM devices every DRAM clock cycle. In 2T timing mode, a new command can be issued to the DRAM devices every other DRAM clock cycle i.e., the address and command bus needs to be held valid for 2 cycles.

2T timing reduces the efficiency of the command bus by half but it doubles the setup and hold time for the command and address bus. The timing for the DRAM data bus and all other signals to the DRAM devices remain the same between 1T and 2T timing modes.

On the EP80579, unbuffered dual DIMM configurations are supported in 2T timing mode. Single DIMM configurations are supported in 1T timing mode. Please refer to Table 11-5 for details. The 2T timing mode can be selected by setting the 2T or1T bit in "Offset 64h: DRT1 – DRAM Timing Register 1".

Note: 1T/2T timing is also referred to as 1N/2N timing.

Figure 11-3. 2T and 1T Timing Mode



11.3.5 DQ/DQS Mapping

The data signal (DQ) to data strobe (DQS) relationship is controlled by the setting in the DRAM Row Attribute (DRA) registers (see Section 16.1.1.40, "Offset 70h: DRA[0-1] – DRAM Row [0:1] Attribute Register" for details). Bits 7:6 and 3:2 of these registers respectively define the device width for the odd and even rows, which is also used in the mapping of DQS signals to DQ signals.

Table 11-13. DRA Mapping for DQS

Bits	Definition	DQS per DQ
00	Reserved	NA
01	x8 DDR	1 DQS strobe per data byte
10	Reserved	NA
11	Reserved	NA

Table 11-13 shows the mapping of DQS to DQ in general terms. Table 11-14 gives the exact relationship.

Table 11-14. DQS to DQ Mapping for x8 Devices

	x8 devices
DQ Byte	DQS bit
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8

11.3.6 32-Bit Mode

DQ[31:0] and the associated DQS[3:0] are connected to the DIMMs in 32-bit mode. Operation in 32-bit mode is invisible to software and on-chip memory controller interfaces.

11.4 DDR2 Features

The DDR2 generation of technology introduces some new features beyond standard DDR. This section highlights those supported.

11.4.1 Interface Signalling Voltage

The memory controller supports 1.8 V signaling for DDR2-400, DDR2-533, DDR2-667 and DDR2-800 DIMMs.

11.4.2 On-DIMM Die Termination (ODT)

The JEDEC DDR2 DRAM Specification requires that DDR2 DIMM devices provide selectable on-DieDIMM Termination (ODT) as an alternative to traditional discrete termination on the motherboard. The On-DIMM termination ODT feature is enabled via the DDR extended memory register select (EMRS) command ODTENA and supports both 75 and 150 terminations activated dynamically via dedicated On-DIMM termination ODT interface signals on the DIMM.

The EP80579 implements two ODT pins to control the behavior of the termination on the two ranks of DRAM devices. The mapping of the CS & ODT pins for the various DIMM configurations is shown in Table 11-6. The behavior of these ODT signals can controlled by setting the appropriate bits in the Section 16.1.1.49, “Offset B0h: DDR2ODTC - DDR2 ODT Control Register”.

The EP80579 controls operation of DDR2 devices with or without on-DIMMDie termination enabled, but is verified by Intel only in On-Die (ODT) enabled mode of operation. ODT can be disabled by setting the appropriate bits in DRAMODT bit in Section 16.1.1.43, “Offset 7Ch: DRC – DRAM Controller Mode Register”.

Table 11-15 shows the ODT related timing parameters.

The EP80579 supports operation of DDR2 devices with or without on-DIMMDie termination enabled, but is verified by Intel only in the On-Die (ODT) enabled mode of operation. ODT can be disabled by setting the DRAMODT bit in Section 16.1.1.43, “Offset 7Ch: DRC – DRAM Controller Mode Register”.

Table 11-15. ODT Timing Parameters

Parameter	Description	Value
tCK	Clock Period	5, 3.75, 3, 2.5ns
tOND	ODT turn on delay	2 tCK
tOFD	ODT turn off delay	2.5 tCK
CL	CAS Latency	3, 4, 5, 6
RL	Read Latency	CL
WL	Write Latency (RL - 1)	CL - 1
BL	Burst Length	4, 8
ODT_RD_on	Assertion of ODT pin to inactive slot during reads	= RL - tOND - 1 = RL - 3
ODT_RD_ontime	Time for which the ODT pin is asserted	= tOND+1+BL/2+0.5-tONF = 2+1+BL/2+0.5-2.5 = BL/2+1
ODT_WR_on	Assertion of ODT pin to inactive slot during writes	= WL - tOND - 1 = WL - 3
ODT_WR_ontime	Time for which the ODT pin is asserted	= tOND+1+BL/2+0.5-tONF = 2+1+BL/2+0.5-2.5 = BL/2+1

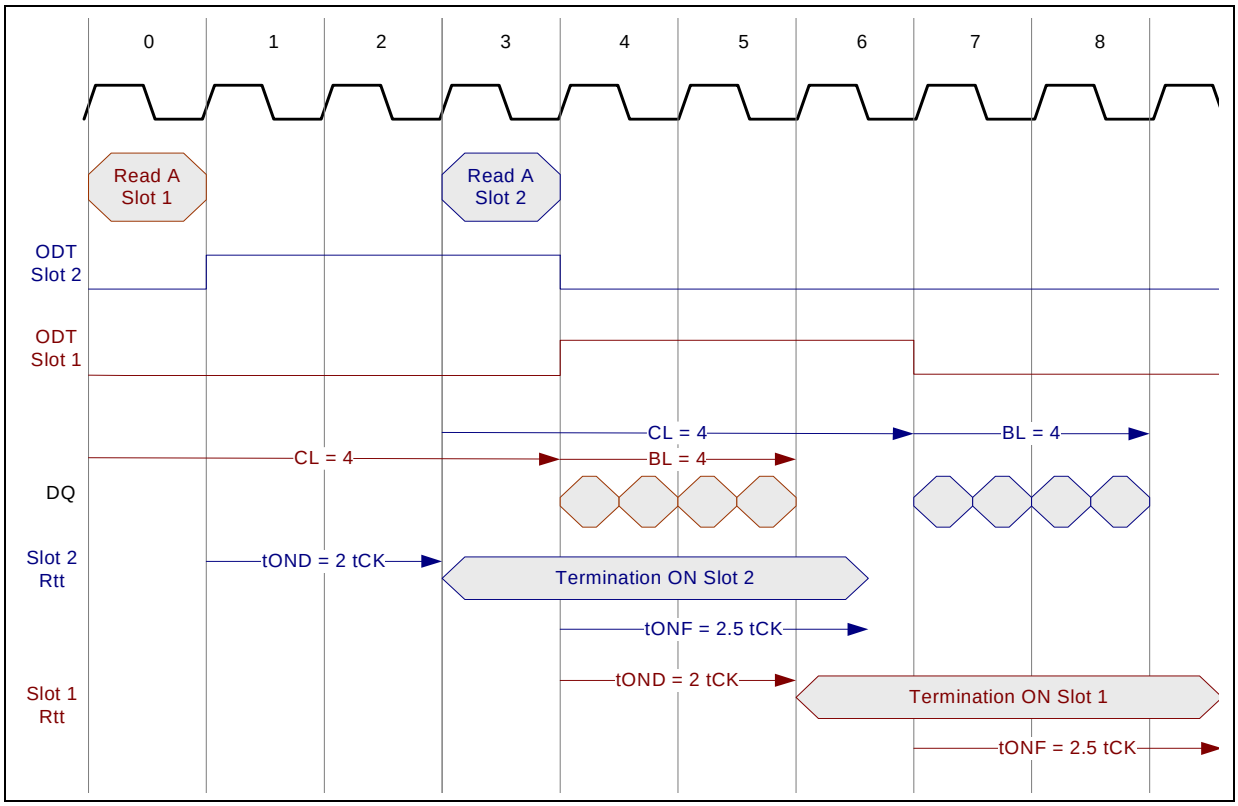
- For WL = 2 (CL = 3), the ODT_WR_on = -1. This requires the controller to enable ODT 1 tCK before the Write command is issued. EP80579 memory controller will not support asserting the ODT pin before it issues the write command. The result is that for CL=3, the termination in the inactive slot will be turned on at the same time the DQ bus is being driven by the memory controller.
- The ODT turn on and off timings will be calculated by the hardware based on the DRAM configuration parameters: CL and BL.

11.4.2.1 ODT Control of Reads

During a read command in a 2 slot configuration as shown in Figure 11-4, the memory controller will enable ODT on the inactive slot. The ODT pin will be asserted by the controller such that the termination in the inactive slot will be enabled 1 tCK before the active slot starts driving the DQ bus. The termination will be turned-off by the controller ½ tCK after the last DQ is driven by the active slot as show in Figure 11-4.

There is no assertion of the ODT signal to the DRAM device in a 1 slot configuration. Please refer the Section 16.1.1.49, "Offset B0h: DDR2ODTC - DDR2 ODT Control Register" for more details on the CSR that needs to be programmed by BIOS so that the memory controller drives the proper ODT control signals.

Figure 11-4. ODT Timing on Back-to-Back Reads to Different Slots



11.4.2.2 ODT Control of Writes

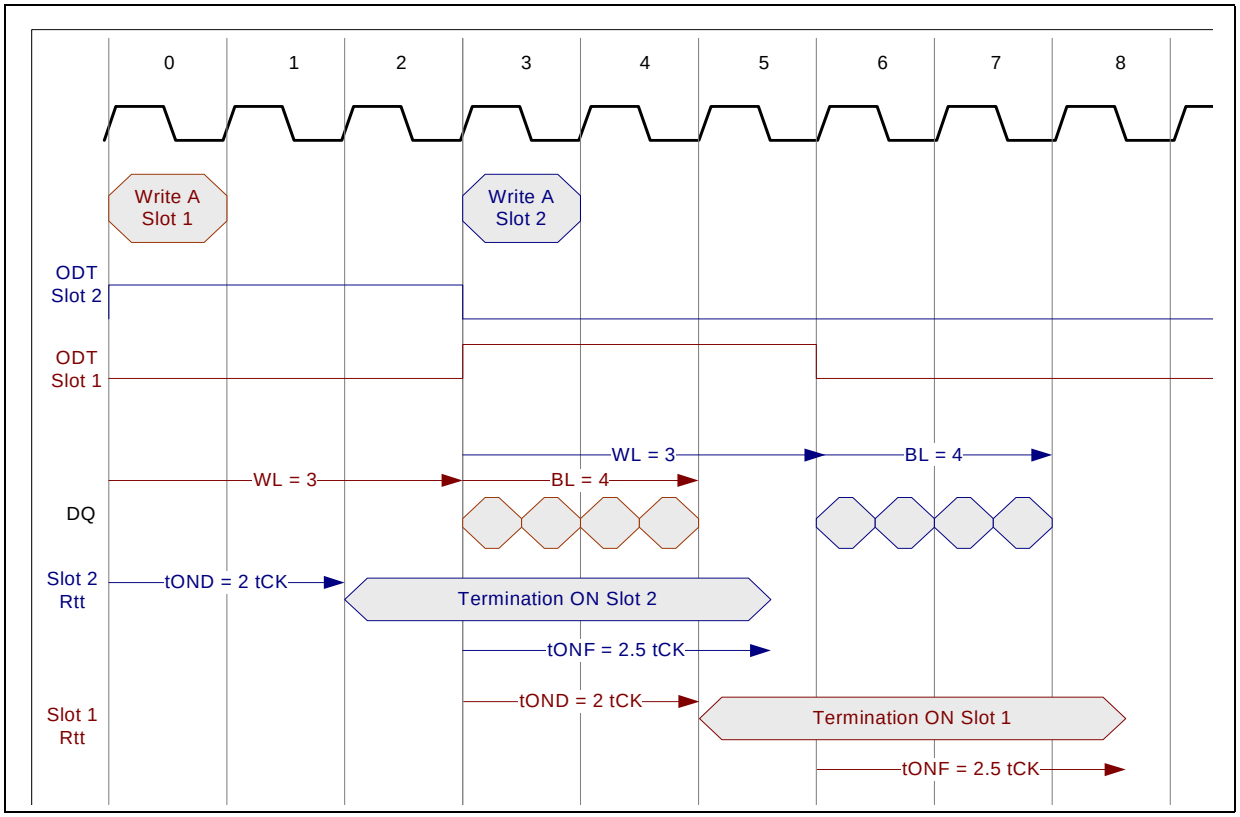
During a write command in a 2 slot configuration as shown in Figure 11-5, the EP80579 memory controller will enable ODT on the inactive slot. The ODT pin will asserted by the controller such that the termination in the inactive slot will be enabled 1 tCK before the controller starts driving the DQ bus (except for configurations where CL = 3). The termination will be turned-off by the controller ½ tCK after the last DQ is driven by the controller to the active slot as shown in Figure 11-5.

During a write command in a one slot configuration, the memory controller will enable ODT to the active slot such that the termination is enabled 1 tCK before the controller starts driving the DQ bus (except for configurations where CL = 3, see note below).

Please refer the Section 16.1.1.49, “Offset B0h: DDR2ODTC - DDR2 ODT Control Register” for more details on the CSR that needs to be programmed by BIOS so that the memory controller drives the proper ODT control signals.

Note: The memory controller will not enable ODT before the write command is issued on the DRAM interface (See Table 11-15 for more details). For configurations with CL=3, WL=2 the controller will assert the ODT pin along with the write command. This will result in the termination in the inactive slot to be enabled at the same time the controller starts driving the DQ data bus.

Figure 11-5. ODT Timing on Back-to-Back Writes to Different Slots



11.4.3 On-Die Termination (ODTZ) on the EP80579

The EP80579 supports ODT (referred to as ODTZ to differentiate it from the On-DIMMDie Termination, ODT that is implementation on the DRAM devices) on the DQ/DQS buffers to improve signal integrity on the inbound path (read data from DRAM).

ODTZ will be enabled on read accesses to the DRAM devices and can be in one of the 3 states - 60ohms, 120ohms or off. ODTZ will be automatically disabled when the EP80579 DQ/DQS buffers are enabled for a write access to the DRAM device. There are no timing parameters implemented to control this functionality and the DDRIO pads are responsible for ensuring that ODTZ is in the disabled state when issuing writes to the DRAM devices.

The termination value for ODTZ can be set by programming the ODTZENA bit in Section 16.1.1.45, “Offset 88h: SDRC – DDR SDRAM Secondary Control Register”.

11.4.4 Refresh

The EP80579 supports generation of the refresh commands (REF) that are necessary for the DRAM devices to retain its data. When the refresh cycle is launched by memory controller an address counter, internal to the DRAM device, supplies the bank address during the refresh cycle. No control of the address bus is required for a refresh cycle. When the refresh cycle has completed, all banks of the DDR2 device will be in the precharged (idle) state. A delay between the Refresh command and the next activate command or subsequent Refresh command must be greater than or equal to the Refresh cycle time (tRFC). There are 3 mechanisms through which the EP80579 memory controller will generate the refresh commands:

1. **Programmable counter:** The refresh engine can be programmed to generate refresh commands at programmable time intervals. The choice of intervals are meant to cover DDR2 device tREFI specifications. Please refer to Section 16.5.1.3, “Offset 40h: DCALCSR – DDR Calibration Control and Status Register” details on programming the refresh engine.
2. **Programmable OPCODE generation:** A single refresh command can be generated under software control using the available opcodes in Section 16.5.1.3, “Offset 40h: DCALCSR – DDR Calibration Control and Status Register”.
3. **Self-Refresh exit state machine:** The self-refresh exit engine will issue one refresh command to each rank after it brings them out of self refresh.

Note: The EP80579 does not support posted refresh cycles.

11.4.5 Self-Refresh

The EP80579 supports the generation of self-refresh commands that can be used to retain data in the DRAM devices without any support from the memory controller. The DRAM device has built-in counters timers to accommodate the self-refresh operation.

There are 2 mechanisms to enter and exit the self-refresh mode:

1. Self-Refresh Entry
   a. **S3:** The memory controller will issue a self-refresh entry command at the end of the S3 sequence.
   b. **Programmable OPCODE generation:** A self-refresh entry command can be generated under software control using the available opcodes Section 16.5.1.3, “Offset 40h: DCALCSR – DDR Calibration Control and Status Register”.
2. Self-Refresh Exit
   a. **Power up after S3 event:** The memory controller implements a self-refresh exit engine which under software control can bring the DRAM devices out of self-refresh. Refer to Table 16-226, “Rules about issuing Self-Refresh and Refresh commands using DCALCSR.OPCODE” on page 604 for details on the rules that software should follow when using this mechanism.
   b. **Writing to DRC.CKE[1:0] register bits:** By writing to the DRC.CKE[1:0] registers, software can assert the CKE pins to the DRAM devices bringing them out of self-refresh.Please refer to Section 16.5.1.59, “Offset 1F4h: MB_ERR_DATA32 - Memory Test Error Data 3” for more details.

The de-emphasis feature on the command/clock pins should be disabled before entering self-refresh. Please see the DEMCA bit in Section 16.5.1.63, “Offset 264h: DDRIOMC1 - DDR IO Mode Control Register 1” for more details.

11.4.6 RCOMP

The EP80579 supports RCOMP for the DDR pads. RCOMP control is implemented for 2 groups of DDR pads. The RCOMP used for the 2 groups can either be static or dynamic based on DDRIOMC2.LEGOVERRIDE[5:4]. For more details on how to control the RCOMP for the 2 groups of DDR pads please see Section 16.5.1.64, "Offset 268h: DDRIOMC2 - DDR IO Mode Control Register 2". The final values that are used for Group 1 and Group 2 drivers can be viewed using the bits in Section 16.5.1.29, "Offset F0h: DIOMON - DDR I/O Monitor Register".

- Group1: Data, data mask and data strobes:
  - DDRIOMC2.LEGOVERRIDE[4:0] bits can be used in dynamic mode to achieve the target impedance on the group 1 DDR pads. In dynamic RCOMP mode, these CSR bits control several pull-up devices that are binary sized. The pull-up devices are tuned using an external resistor (Rext). In Static mode, DDRIOMC2.LEGOVERRIDE[3:0] determines the RCOMP value used by the Group 1 DDRIO pads.
- Group2: Command, Address and Clock:
  - In dynamic RCOMP mode the digital control for these group of DDR pads is derived by multiplying the group 1 digital control by a factor determined by DDRIOMC2.LEGOVERRIDE[6:9]. In static RCOMP mode, DDRIOMC2.LEGOVERRIDE[6:9] directly controls the RCOMP value used by the Group 2 DDRIO pads.

11.4.7 DDR2 MR and EMR settings

Table 11-16 shows the supported settings of DDR2 mode register (MR) and extended mode register (EMR). Note that only the architecturally relevant settings of the MR and EMR are listed in this table. These registers can be updated using the MRS and EMRS commands. Please see Section 16.5.1.3, "Offset 40h: DCALCSR – DDR Calibration Control and Status Register" and Section 16.5.1.4, "Offset 44h: DCALADDR - DDR Calibration Address Register" for more details on generation of MRS and EMRS commands.

Table 11-16. Supported DDR2 MR and EMR settings

MR/EMR Feature	EP80579 Support
Burst Type	Sequential only
Burst Length	4 (64-bit mode) and 8 (32-bit mode)
Write Recovery for Auto-precharge	Supported. Settings based on speed bins. Please see Section 16.0, "IMCH Registers" for details.
CAS Latency	3, 4, 5 and 6 depending upon Speed bin. Please see Table 11-12 and Section 16.0, "IMCH Registers" for more details.
ODT (EMR)	Supported. Please see Section 11.4.2 for more details.
OCD Calibration (EMR)	Not Supported.
Additive Latency (EMR)	Not Supported.

11.4.8 Scrubbing Support

The memory controller will support both demand scrubbing and background scrubbing in hardware. This section provide details of the scrubbing mechanism.

11.4.8.1 Demand Scrubbing

When the controller detects a single bit error on a DRAM read, the ECC logic fixes the single bit error and returns the corrected data to the requester. The controller also writes the corrected data back into DRAM using the demand scrub operation.

11.4.8.2 Background Scrubbing

The controller supports a simple state machine that periodically injects read transactions into the stream. The scrub rate meets a general requirement for scrubbing a 4GB memory capacity every 24 hours. It can also be programmed to scrub the entire DDR at faster rates, if that is desired. Rates available extend from an entire scrub every few days, to every day, every hour, every minute, and even very fast rates, mainly used for validation purposes. Data for a background scrub read operation is discarded if no error is detected on the read. When a single bit error is detected on a background scrub read transaction, the mechanism specified in Figure 11.4.8.1 is used to correct the single bit error in DRAM. The read data is dropped. When a double bit error is detected on a background scrub read transaction, the controller log the error and interrupts the IA-32 core.

11.5 Error Handling

The memory controller accesses may encounter data with ECC violations and/or parity errors.

- For writes with bad parity, as indicated on bits in the command from either AIOC memory target (MT) or IMCH, the memory controller will execute the write, poisoning the data as it writes it to DDR. Please see Table 11-17 for the granularity of poisoned data. Poisoning is accomplished by inverting each bit of ECC calculated for that particular write, based on the bad write data sent to the memory controller from AIOC or IMCH.
- For reads from DDR with single bit parity errors, the memory controller will correct all correctable errors, and return the data, to either requestor, corrected, and without a bad parity indication. Status logging, as discussed above, will be done.
- For reads from DDR with multiple bit, uncorrectable ECC violations, the memory controller will return the data to either requestor, however, a bad parity indication will be driven back with the data. Please see Table 11-17 for the granularity of poisoned data. AIOC memory target (MT) or IMCH should poison the data when seeing the bad parity bit set, along with the read data returned.

Table 11-17. Poisoning Granularity

Command	Error Source	Command Source	Memory Controller Action
Read from DRAM	DRAM (DED)	IMCH	Poison parity bits on 32B granularity and return read data to IMCH.
Read from DRAM	DRAM (DED)	AIOC MT	Poison parity bits on 8B granularity and return read data to AIOC memory target (MT).
Write to DRAM	IA or IO Device including AIOC (Parity Error)	IMCH	Poison ECC bits on 16B granularity and write to DRAM.
Write to DRAM	AIOC (Parity Error)	AIOC MT	Poison ECC bits on 8B granularity and write to DRAM.
• ECC can be enabled by setting the DDIM bit in Section 16.1.1.43, "Offset 7Ch: DRC – DRAM Controller Mode Register". • Poisoning of write data to DRAM can be enabled using the MEMPEN bit in Section 16.1.1.44, "Offset 84h: ECCDIAG – ECC Detection/Correction Diagnostic Register". • Poisoning of read data from DRAM to IMCH or AIOC MT can be enabled by using ENDP bit in Section 16.1.1.43, "Offset 7Ch: DRC – DRAM Controller Mode Register".			

For compatibility with accepted IA platform algorithms and mechanisms, the memory controller will follow general IA error logging and reporting mechanisms as closely as possible with the following exception:

- The memory controller does not implement uncorrectable retries (DED retries). Therefore all registers and bit definitions that support uncorrectable retries are not implemented in the memory controller.

§ §

12.0 Enhanced Direct Memory Access Controller (EDMA)

The IMCH includes an integrated four-channel Enhanced Direct Memory Access (EDMA) controller to facilitate “push model” block transfers without IA-32 core intervention for higher overall system performance. This section details the operating modes, setup, interfaces, register set, and high-level implementation of the EDMA controller.

12.1 Overview

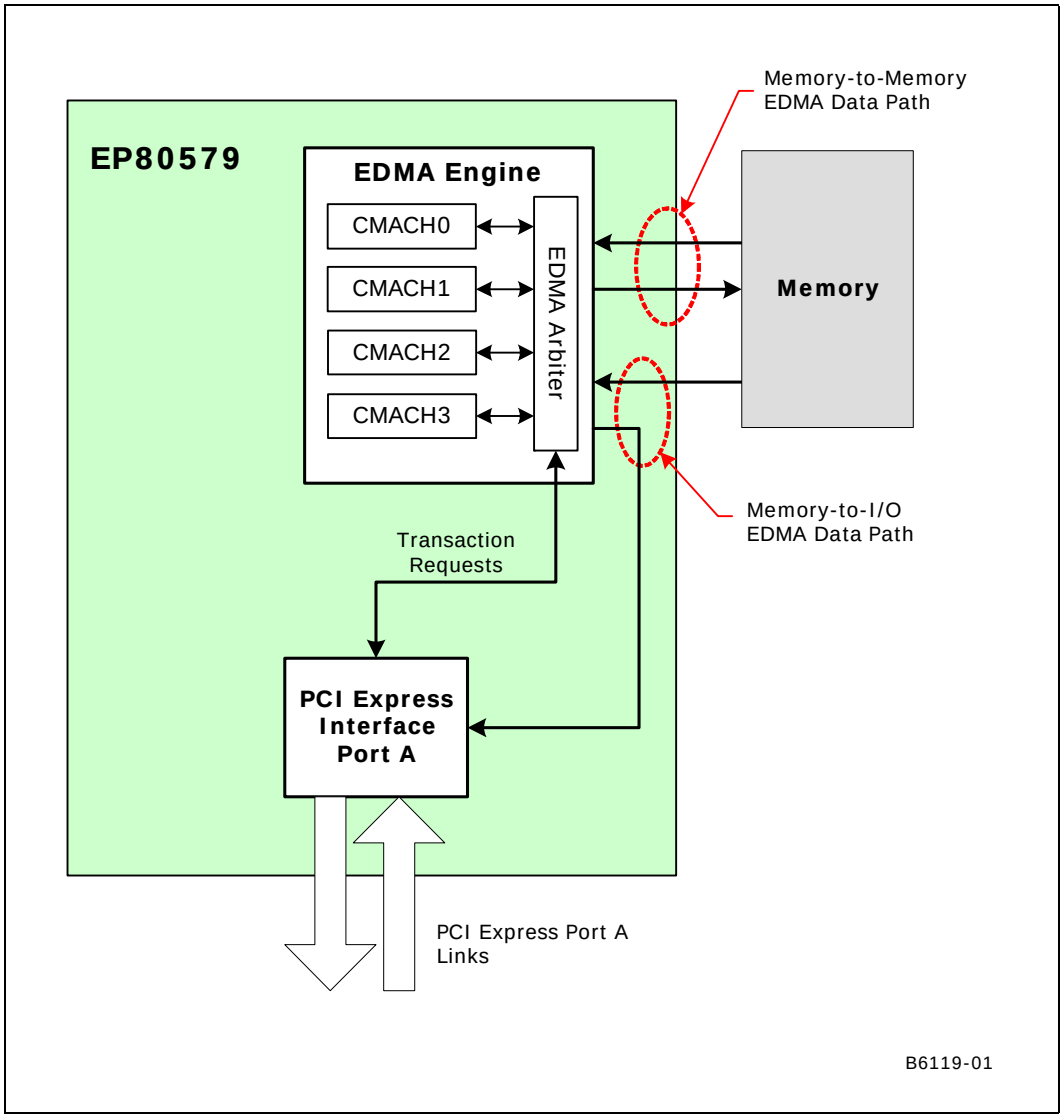
The EDMA engine provides a highly efficient means to move data within local system memory or from the local system memory to the I/O subsystem. Each EDMA channel provides low-latency, high-throughput data transfer capability with minimal IA-32 core intervention. For the IA-32 core, it is a “fire and forget” type of memory transfer, with a doorbell starting mechanism and interrupt capability for signaling completion.

Each channel optimizes block transfers of data through a linked-list descriptor chaining mechanism that supports scatter/gather operations. Each channel is responsible for providing the EDMA programming interface, executing the data transfers, and handling any errors encountered during operation.

Each channel initiates traffic on both the local system memory and outbound traffic arbiter interfaces, and is designed such that each independent channel is capable of generating at least 1 GB/s of traffic during data hauls. In the absence of competition from other traffic sources, multiple channels could theoretically saturate the local memory interface. See Figure 12-1.

Each channel is independently enabled by setting the Start bit in the Control Configuration Register. The Start bit is cleared after power-up or reset and consequently the EDMA controller is disabled until software explicitly turns each one on.

Figure 12-1. Concept Diagram of EDMA Data Path



12.1.1 Features

The following features are supported by the EDMA controller:

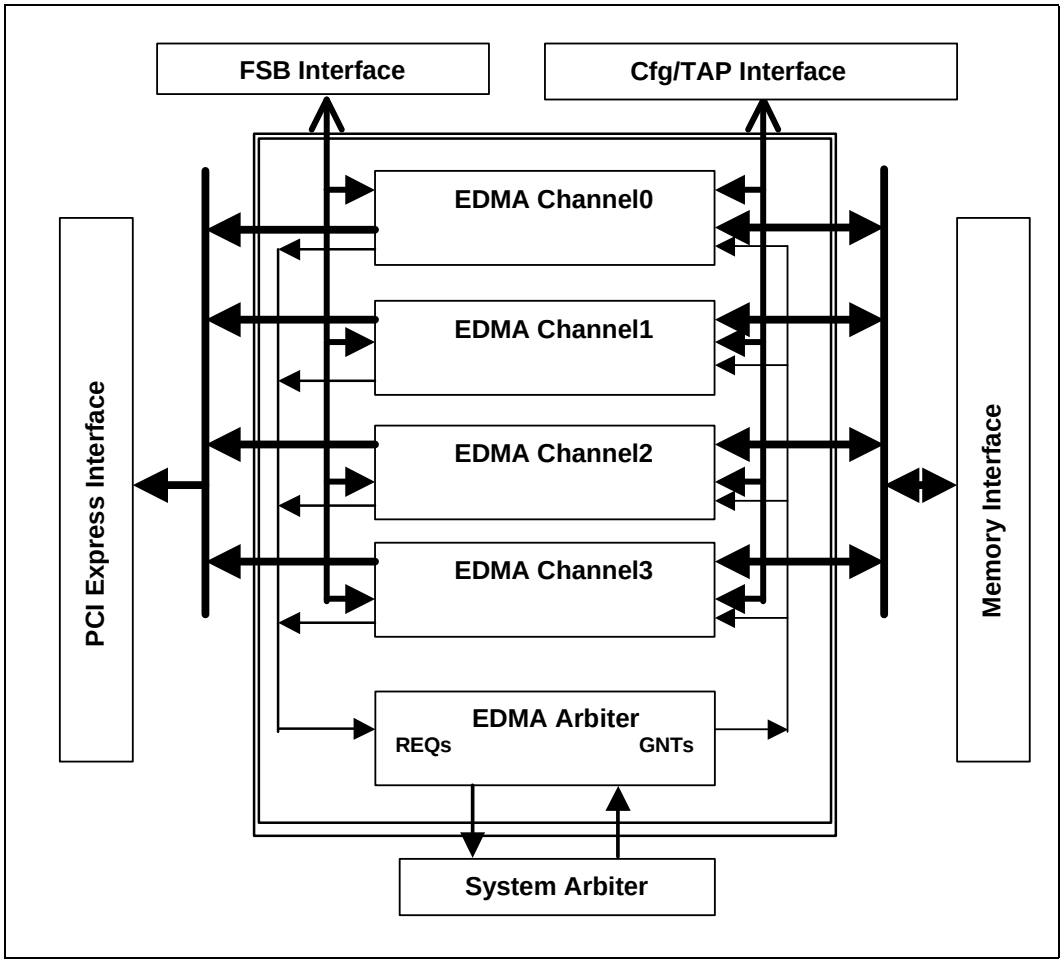
- Four independent channels (see Figure 12-2)
  - — Dedicated data transfer queue per channel
  - — Full register set for descriptor and transfer handling per channel
- Support for transfer between main memory locations, and from memory to the I/O subsystem
- PCI Express* support of traffic class to provide external prioritization of traffic
- Supports transfers only between two physical addresses
  - — 32-bit (4 GB) addressing range on the local system memory interface

  - — 32-bit addressing range on the Memory Mapped I/O Subsystem Interface (no NSI access)
- Maximum transfer of 16 MB transfers per descriptor
- Fully programmable by the IA-32 core
  - — Configuration space mapping for EDMA engine capability and control
  - — Memory-mapped space for EDMA channel-specific register sets
- Chain Mode EDMA transfer with automatic data chaining for scattering/gathering of data blocks
  - — EDMA chaining continued until a “null” descriptor pointer is encountered
  - — Support for appending a block to the end of current EDMA chain
  - — Automated descriptor retrieval from local memory during chaining – single read
- Programmable independent alignment between source and destination
  - — Byte aligned transfer on the local system memory interface
  - — Byte aligned transfer on the I/O subsystem interface
- Support for non-coherent transfers both to and from system memory on a per descriptor basis
  - — Independent control of coherency for source and destination
- Programmable support for interrupt generation on block–by-block basis
  - — Selectable MSI or legacy level-sensitive interrupt function
  - — End of current block transfer
  - — End of current chain
  - — For any error causing a transfer to abort
- Increment of the source and destination address for standard transfers
- Increment of the destination and decrement of the source address to enable byte stream reversal
- Constant address mode for the destination address based on the transfer granularity to enable targeting of memory mapped I/O FIFO devices
- Buffer/memory initialization mode

12.1.2 Logical Block Diagram

Figure 12-2 shows the conceptual interface of the EDMA channels to different I interfaces.

Figure 12-2. Conceptual Diagram of Four Channel EDMA Engine



12.2 Channel Programming Interface

The EDMA channel programming interface is accessible from the IA-32 core via a combination of chain descriptors (shown in Figure 12-3) written to main memory and a memory-mapped internal register set. The EDMA controller provides four channels, each of which can be independently used to transfer data within the local system memory or from the local system memory to the I/O subsystem. Each channel has its own set of 12 registers. Refer to "Memory Mapped I/O for EDMA Registers" on page 651 for a description of the channel register set.

The channel programming interface is accessible from the IA-32 core via a combination of descriptors written to main memory and a memory-mapped internal register set. Each channel is programmed independently.

Each channel supports full chaining capability. The chain descriptors can be cascaded together in system memory to form a linked list. Each chain descriptor contains all the information necessary for transferring a block of data, as well as a pointer to the next chain descriptor in the list. The next descriptor pointer of the last chain descriptor in a linked list will be a null pointer (address zero), indicating the end of that chain.

Throughout this document, all register references are made using the name of the lower 32-bit register, irrespective of whether the target is a 32-bit or 64-bit register with lower and upper halves.

12.3 Chaining Operation

An EDMA access transfers a block of data from one address to another. The desired transfer is specified by setting up a linked list of chain descriptors in the local system memory, and initiated by programming the first chain descriptor start address into the Next Descriptor Address Registers (NDAR/NDUAR) of the EDMA channel and setting the Start bit of the Channel Control Register (CCR). Each block of the transfer is defined by a descriptor in main memory containing the source address, destination address, transfer length and control values. Setting the Start bit of the Channel Control Register (CCR) causes the channel to fetch the current chain descriptor information and place it into its corresponding register set. Once all the register information has been fetched, the actual data transfer starts.

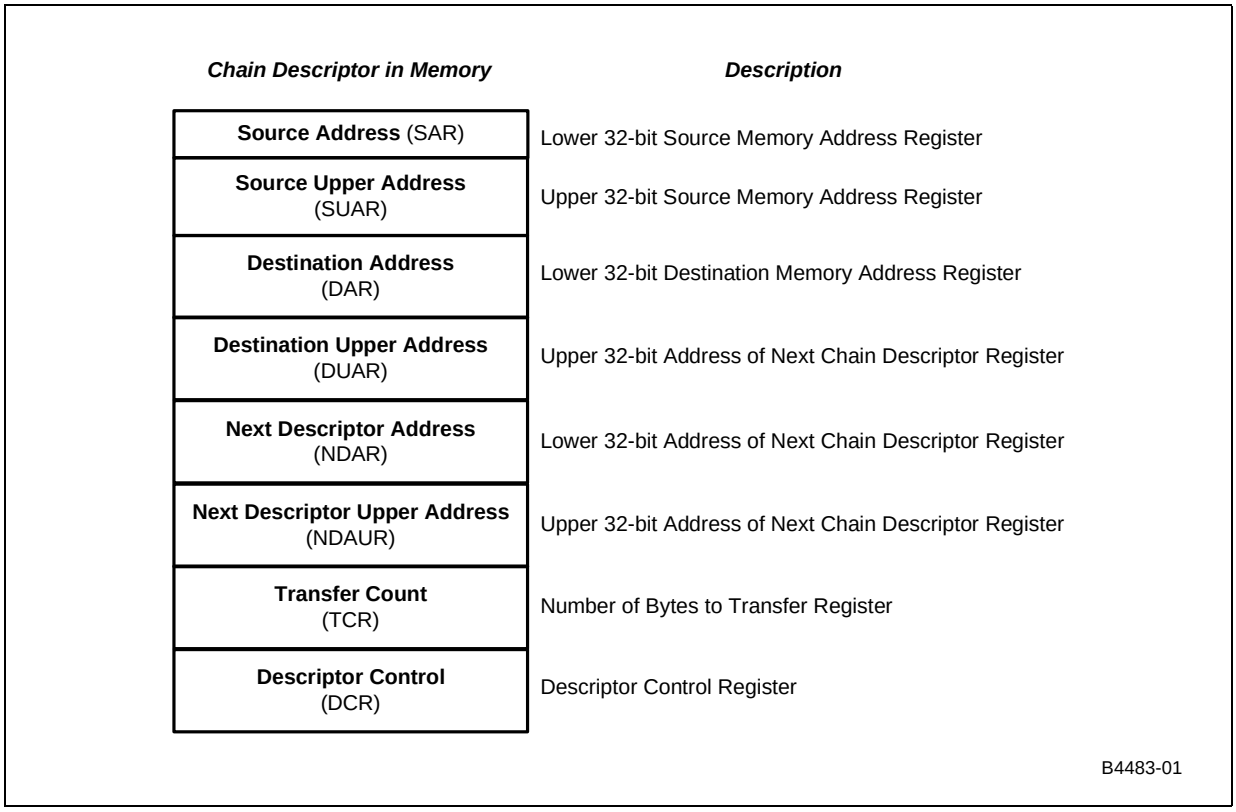
Note: The Start bit will be ignored unless the Channel Status Register (CSR) is in an appropriate state. Software must ensure that the status bits for end of chain, stopped, aborted, and active are all clear prior to attempting to initiate a new transfer with the start function.

12.3.1 Chain Descriptor Definition

All EDMA transfers are controlled by chain descriptors in the local system memory. A single block transfer will specify only a single chain descriptor. Chain descriptors can be linked together to form a linked list, providing a capability for complex EDMA scatter/gather operations.

Figure 12-3 shows the format of a chain descriptor. Each chain descriptor consists of eight contiguous DWords (32-bits) in the local system memory, and must be naturally aligned to an eight Dword boundary. All eight DWords must be defined and are required for the proper operation of the EDMA engine.

Figure 12-3. Chain Descriptor in Memory



12.3.2 DMA Chain Descriptor in Memory

Each chain descriptor is composed of eight Dwords. Each Dword in the chain descriptor in the local memory is analogous to the corresponding EDMA channel register value. The bit definitions in the chain descriptor in memory are identical to those of the corresponding channel register. Refer to the EDMA channel-specific register definitions in "Memory Mapped I/O for EDMA Registers" on page 651 for descriptions of the fields defined by a chain descriptor.

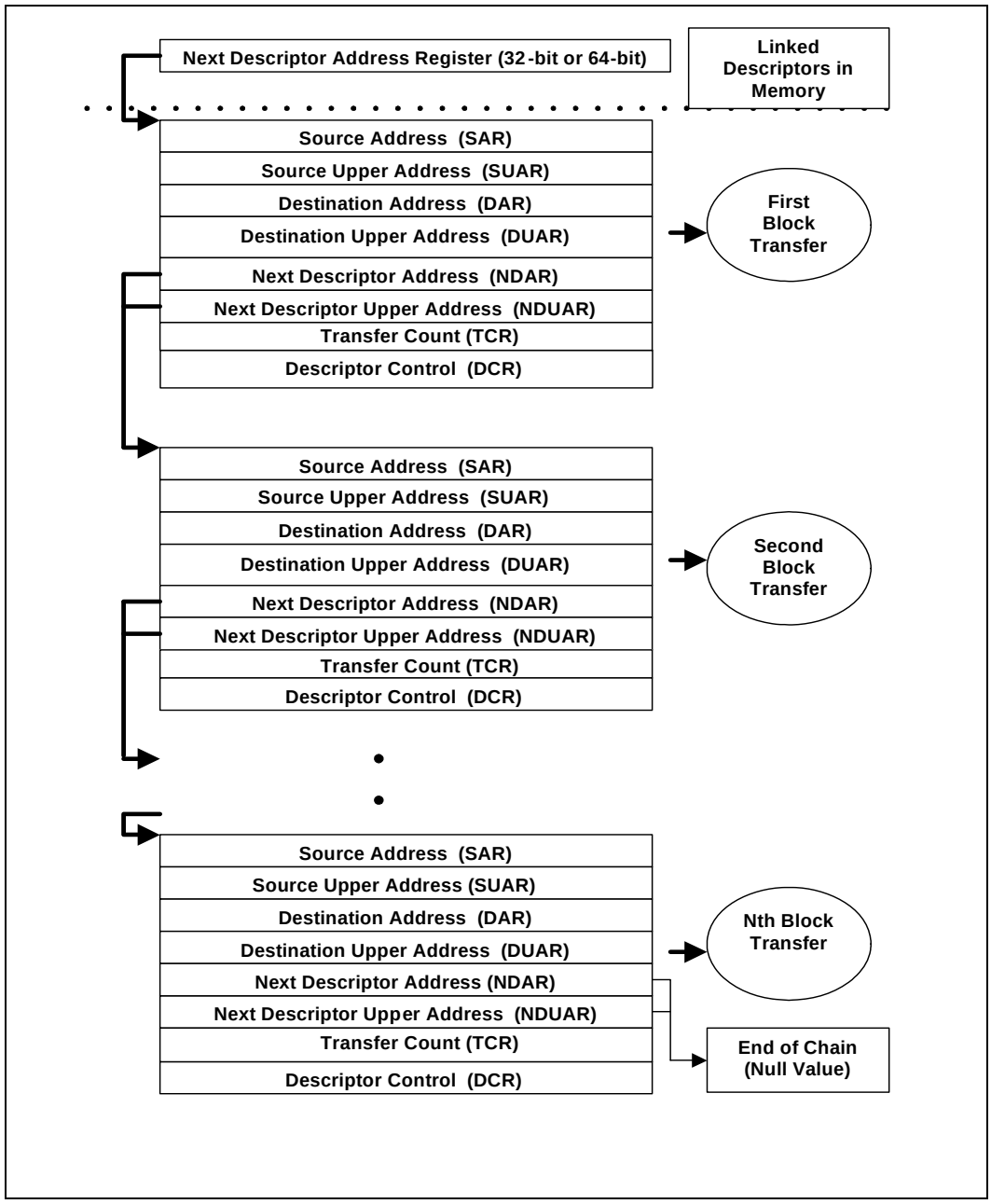
After a transfer has been requested, the EDMA channel reads the specified chain descriptor from local system memory and updates its own corresponding channel registers automatically.

12.3.3 Chain Descriptor Usage

A linked list of chain descriptors may be built in the local system memory to transfer data within the local system memory or from local system memory to I/O subsystem memory. An application may build multiple chain descriptors to transfer many blocks with differing source addresses, destination addresses, data transfer counts, alignments, and coherence attributes. The application may connect these chain descriptors using the Next Descriptor Address in a sequence of chain descriptors, creating a linked list of EDMA transfers, all of which may complete without any processor intervention. Figure 12-4 shows a linked list of transfers built in local system memory and illustrates how they are "chained" together.

It is possible but unexpected that the source and destination address ranges defined by chain descriptors may overlap in physical memory. While there are scenarios where this may produce no errors in the resulting memory image, the software must ensure that no failure results from such usage. Hardware checks are not built into the EDMA mechanism to ensure that source and destination physical address ranges do not overlap. Similarly, there are no hardware interlocks to ensure that independent channels are not programmed to modify the same address range simultaneously. If software were to create such a situation, the resultant memory image would be indeterminate, since there are no guarantees as to the relative access ordering among simultaneously active channels.

Figure 12-4. Chaining Mechanism



12.3.4 Scatter/Gather Transfer

The EDMA descriptors in memory may be defined such that they cause the channel to perform typical scatter/gather data transfers. To “gather” data, software may create a linked list of descriptors that will move non-contiguous source blocks of data into a contiguous set of destination blocks. To “scatter” data, software may create a linked list of descriptors that will move contiguous blocks of source data to non-contiguous destination blocks. It is even possible to program the EDMA transfer descriptors in such a way that some blocks of source data within a single chain are moved to new memory locations, while other blocks are moved out to the I/O subsystem.

There is no hardware restriction limiting the nature of source and destination address ranges, other than that the source and destination types (memory or I/O subsystem) must match the descriptor address mappings. The IMCH aborts the EDMA operation and reports a programming error if this requirement is not met.

12.3.5 Appending to a Descriptor Chain

After an EDMA channel has started processing a linked list of descriptors, the application software may need to append a chain descriptor to the end of the current chain without tearing down the transfer in progress. Such an operation requires a mechanism to guarantee that a descriptor is never in the process of modification by the IA-32 core *while being retrieved* by the EDMA channel (This would result in a hybrid descriptor loaded into the corresponding EDMA channel register set, and spurious operation). The suspend function of the EDMA controller is defined to facilitate this type of usage.

The preferred mechanism for appending to a linked list currently being processed is to suspend the current transfer, modify the terminal chain descriptor to update its Next Descriptor Address field(s), and then allow the transfer to resume. This is accomplished by first setting the Suspend bit in the Channel Control Register (CCR) followed by a read of the Channel Status Register (CSR) to verify that the suspend has taken effect.The Next Descriptor Address fields of the terminal chain descriptor in the current linked list are updated with the address of the first descriptor to be appended. After this update has occurred, software then clears the Suspend bit, sets the Channel Resume bit and allows execution to proceed.

Note: A single write to the CCR may update both bits simultaneously.

This append algorithm covers the following cases:

- The EDMA channel has completed execution of the terminal descriptor in the original chain, and is idle. The EDMA channel examines the Channel Resume bit when the Channel Control Register (CCR) is written. If the bit is set, the EDMA channel will automatically clear the bit and re-read the last chain descriptor (as indicated by CDAR/CDUAR), which updates NDAR with the appended chain descriptor address. A non-null value in NDAR will result in a fetch of the target chain descriptor, and resumed execution. (If the resulting NDAR/NDUAR pair remains null, the EDMA channel will remain idle.)
- The EDMA channel is executing a descriptor prior to the terminal chain descriptor in the linked list. Regardless of whether the channel completes execution of its current descriptor prior to the CCR write to clear Suspend and set Channel Resume, the channel will re-read the current chain descriptor in response to the Channel Resume bit. The next chain descriptor in the chain will be fetched from the address indicated by NDAR/NDUAR, and the channel will continue execution. The appended chain descriptor (or descriptors) will be executed after the channel reaches the end of the original chain.
- The channel is executing the terminal chain descriptor at the time of the suspend command. The channel will complete the final chain descriptor of the original linked

list and examine the state of the Channel Resume bit when the Suspend bit is cleared. As in the prior case, the channel will re-read the current chain descriptor to update NDAR/NDUAR, load the first appended chain descriptor, and resume execution. The only difference in this case is that the re-read operation on the current chain description was required for proper execution (in the prior case it was wasted effort, but did not result in erroneous behavior). If the channel had completed execution of the terminal chain descriptor and set the “end of chain” status bit, this bit is automatically cleared when the channel resumes operation.

- If the current transfer had a non-fatal error, it follows one of the above cases. If the error is fatal, the channel will abort, and the software must take proper action and restart the EDMA transfer. Note that the channel will ignore the state of the Channel Resume bit if the abort status has not been cleared from the CSR. This simplifies the case of linked list append, as software need not take extra steps to verify that no errors exist prior to setting the Channel Resume bit. The normal polling or interrupt mechanism may handle the error without interacting with the append routine.

Note: Software is at liberty to modify the Next Descriptor Address fields of the terminal chain descriptor at any time after setting the Suspend bit in the CCR – there is no requirement that software verify that the channel has gone idle prior to modifying the memory image. Also, software does not need to verify that the channel has completed execution of the current chain descriptor and acknowledged Suspend EDMA prior to issuing the final update to CSR that sets the Channel Resume bit. The hardware interlock will cover the case where the end of the chain descriptor is reached during the append sequence, but proper operation is guaranteed regardless of whether the interlock is exercised.

A further simplification to the linked list append sequence is possible in the case of chain descriptors located strictly below the 4 GB boundary in memory; that is, in the case where NDUAR of the terminal descriptor is zero and only NDAR contains asserted bits. Under these conditions, it is safe to issue the NDAR write cycle without first suspending operation, because there is no risk of a hybrid NDAR/NDUAR pair retrieved by the channel. If desired, software could take the simplified approach of issuing the descriptor update followed by a CCR write to set the Channel Resume bit. In all cases, this will result in successful execution of the appended chain irrespective of current execution status.

12.3.6 Splicing a Descriptor Chain into a Linked List

Software may utilize a slight modification of the algorithm described in “Appending to a Descriptor Chain” on page 314 to splice a new descriptor or chain of descriptors into the chain already executing. Such an operation would be useful to provide service to a higher priority EDMA transfer without aborting work already in progress.

The steps required to splice into a chain are as follows:

1. Write to the CCR to set the Suspend EDMA bit.
2. Read the CDAR/CDUAR pair to determine which chain descriptor the EDMA channel is currently executing.
3. Read the Next Descriptor Address field of the current chain descriptor, and write the retrieved address into the Next Descriptor Address field of the terminal chain descriptor in the linked list to be spliced-in.
4. Write the address of the descriptor (or lead descriptor of the chain) to be spliced-in into the Next Descriptor Address field of the current descriptor (in memory).
5. Write to the CCR to clear the Suspend bit and set the Channel Resume bit.

The hardware interlock of the suspend function will guarantee that the channel will not proceed beyond the current chain descriptor until the Suspend bit has been cleared in the CCR, and the Channel Resume function will guarantee that the current chain descriptor will be re-read to retrieve the modified NDAR/NDUAR value pointing to the spliced chain descriptor. The channel will resume execution with the head of the spliced linked list, and will traverse that linked list back to its original list of chain descriptors.

Note: The channel must refrain from updating CDAR/CDUAR from NDAR/NDUAR in addition to dropping the returned data. Were the channel to update its CDAR/CDUAR values, it would “skip” the entire spliced chain in response to Channel Resume, because software would have spliced in the new descriptor chain at a position “behind” the new value for CDAR/CDUAR in the original chain.

12.4 Transfer Types

The EDMA controller is optimized to perform high throughput data transfers between local memory locations, and from local memory to I/O subsystem memory. Supported transfer types are summarized in the following subsections.

12.4.1 Local Memory to Local Memory

The local memory to local memory transfer will move blocks of data specified by descriptors from one region in main memory to another. The channel control hardware will issue read cycles to the local memory interface using an incrementing or decrementing source address, and place the retrieved data into a channel buffer. It will then issue write cycles back to the memory interface using an incrementing or constant destination address. Each EDMA channel supports pipelining making it possible for a single channel to have multiple read and write cycles active at the same time.

All read requests to memory are a full cache-line in length (64 B), so the EDMA channel must discard data as needed to realign the initial read in a transfer to the alignment specified by the source address. The number of cache-line reads issued by the controller in any given internal arbitration cycle is dependent upon the number of available cache-line spaces in the data queue, and upon the configuration of the inbound/outbound arbiter, but is limited to a maximum of two cache-line requests.

All write requests to memory are also a full cache-line in length, although not all bytes must be enabled for every write. The EDMA channel is responsible for translating the alignment specified by the destination address registers and the destination alignment bit in DCR into a corresponding set of byte enables for the initial write in a transfer. Once the initial alignment has been enforced, the rest of the transfer on behalf of any given descriptor is contiguous.

12.4.2 Local Memory to I/O Subsystem Memory

The local memory to I/O memory transfer will move blocks of data specified by chain descriptors from a source region in main memory to a destination region in the I/O subsystem. The channel control hardware will issue read cycles to the local memory interface using an incrementing or decrementing source address, and place the retrieved data into a channel buffer. It will then issue write cycles to the I/O subsystem using an incrementing or constant destination address. Each EDMA channel supports pipelining for this transfer type as well, thus multiple read and write requests may be outstanding from the same EDMA channel at any given time during a block transfer.

All read and write requests are full cache-line size (64 B), irrespective of alignment and length specified in the descriptors; it is up to the EDMA channel to discard read data and calculate write byte enables to enforce the descriptor alignment specified.

The number of reads issued by each EDMA channel in any given internal arbitration cycle is dependent upon the number of available cache-line spaces in the data queue, and upon the configuration of the inbound/outbound arbiter, but is limited to a maximum of two cache-line requests. The number of writes issued in any given arbitration cycle is dependent upon the number of cache-lines waiting in the data queue, and upon the CCR configuration, but is limited to a maximum of two cache-line requests. The EDMA controller never speculatively issues a write in anticipation of data returning from the memory subsystem.

12.4.3 I/O Memory to Local Memory

The I/O memory to local memory transfer is not supported.

12.4.4 I/O Memory to I/O Memory

The I/O memory to I/O memory transfer is not supported.

12.5 Addressing

Each EDMA is capable of 36-bit addressing on both source and destination interfaces.

Alignment specification is independent for source and destination. Transfers may be specified to be aligned to any byte boundary except in destination constant address modes where the granularity is greater than 1-byte.

Each EDMA channel uses direct addressing for both the source and destination interfaces. There is no internal support for any virtual address translation.

Each EDMA channel will attempt to compensate for misalignment between source and destination. At a minimum, misalignment will result in decreased performance at either end of the transfer, where a second read is required prior to the first write, or vice-versa.

12.5.1 Address Coherence

Each EDMA channel provides support for non-coherent access specification to improve bandwidth and provide more consistent average latency, as well as to free the FSB for simultaneous IA-32 core traffic. The source and destination addresses for each DMA channel may be independently specified on a chain descriptor granularity via bit settings in the DCR to be either coherent or non-coherent. For non-coherent accesses, no FSB snoop cycle is issued on behalf of EDMA memory accesses to snoop processor caches. The software must verify that snoops of IA-32 core caches are not required for proper system operation prior to setting either of the non-coherent bits in any given descriptor. Non-coherent accesses are used for un-cacheable memory regions, or for cacheable regions where software can guarantee no modified state in any IA-32 core cache by some other means.

The non-coherent attribute further implies relaxed posted write ordering as defined by PCI/PCI-X. A non-coherent write may pass coherent posted writes en route to memory. Software should verify that snoops of IA-32 core caches are not required for proper system operation prior to setting either of the non-coherent bits in the DCR field of any given descriptor. Software need not take special steps to accommodate the relaxed ordering behavior, because each channel will only generate a single stream of output per descriptor, and no ordering is defined between competing I/O subsystem traffic sources. Non-coherent access may be used for uncacheable memory regions, or for cacheable regions where software can guarantee the IA-32 core cache state has not been modified by other means.

An example usage model for non-coherent accesses is management of data block areas reserved for use by an EDMA-capable peripheral device, such as a network interface controllers (NIC). The NIC writes data directly into a buffer in memory allocated for the exclusive use of that device. Each EDMA channel would then execute a block transfer from that buffer to an area allocated for IA-32 core use. Both integrity verification and security functions executed by the IA-32 core could follow such a model. As long as software ensures that the IA-32 core never traverses the device-allocated memory, the block transfer could be accomplished using non-coherent source address reads followed by coherent destination address writes.

Note: I/O subsystem destination addresses are always treated as non-coherent or coherent based on the bit setting in the DCR. Setting the destination coherency bit will result in the PCI-Express snoop not required attribute bit being clear, snoop required.

Example: Setting BDF 010 Offset 2C DCRx register bit 7

Destination non-coherent = 0 => Snoop not required attribute bit = 1

Destination coherent = 1 => Snoop not required attribute bit = 0

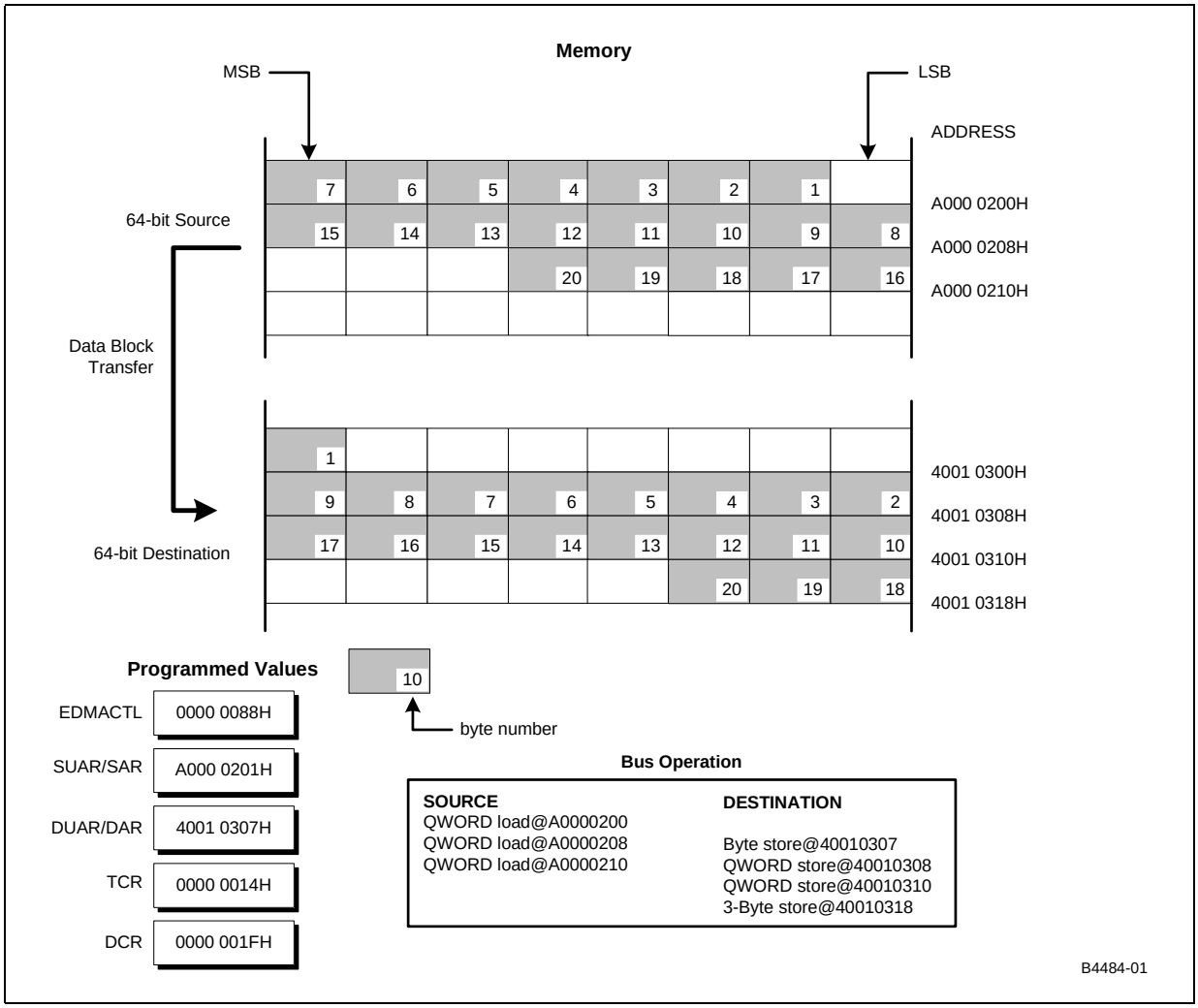
12.5.2 Addressing Modes

Many different addressing modes are available, including standard byte movement mode, byte reversal mode, constant address mode, and memory and buffer initialization modes. In the examples shown for each of the following modes, a 64-bit interface is used for simplicity. The interface could be the memory interface or an external device on an expansion bus. Internally, the EDMA data path is significantly wider.

12.5.2.1 Standard Byte Movement Mode

Standard byte movement mode is the most common method in which data is transferred within the memory sub-system. In this mode, the source and destination are specified down to the byte address. The source address is incremented as data is read and the destination address is incremented as data is written. Transfers can be memory to memory or memory to memory mapped I/O. Figure 12-5 illustrates a memory to memory data transfer between unaligned 64-bit, source and destination addresses.

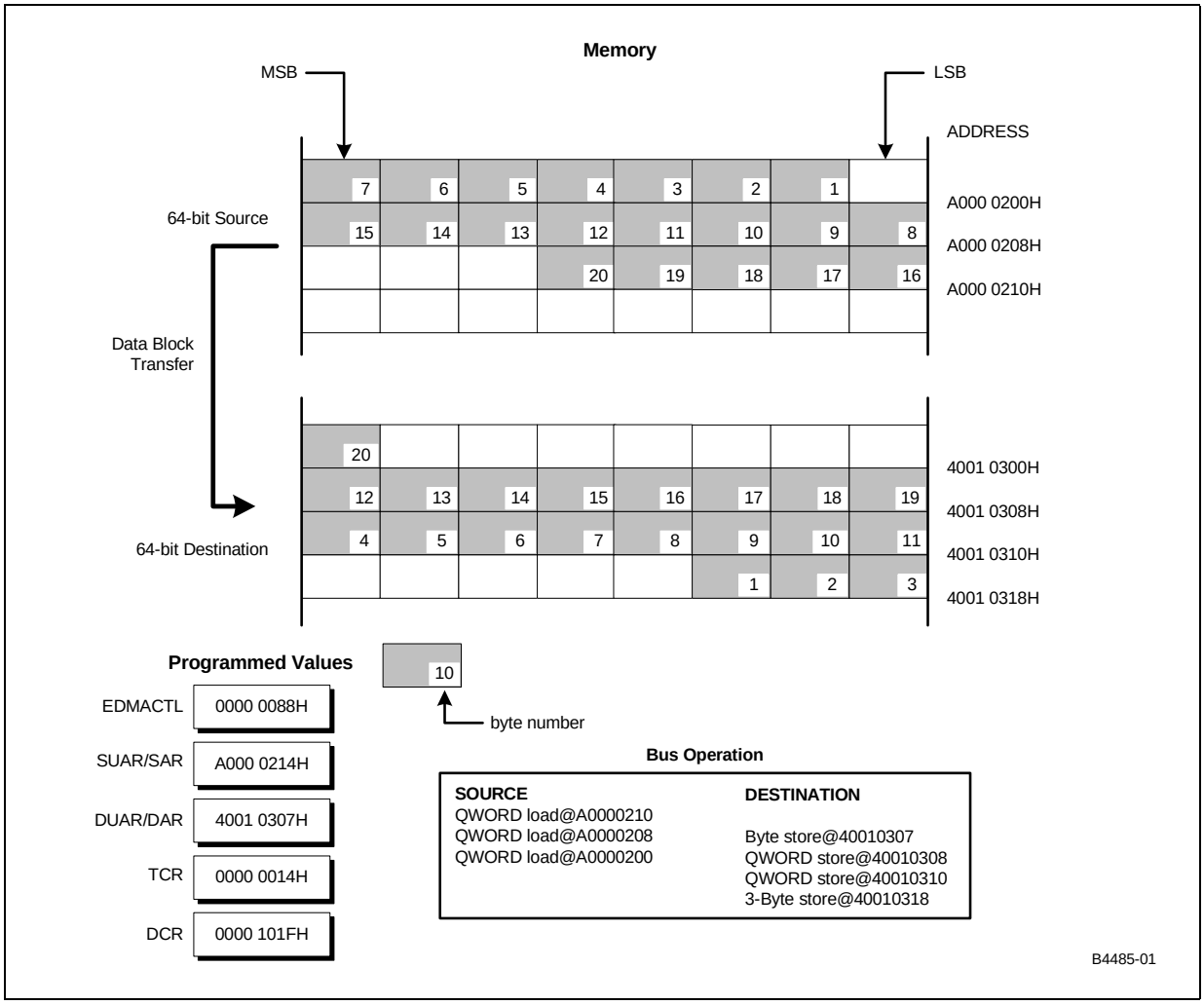
Figure 12-5. Source and Destination in Increment Mode Transfer



12.5.2.2 Decrement/Byte Reversal Mode

Decrement/byte reversal mode is useful when an entire data stream needs to be reversed at the byte level. This must not be confused with endian swapping, as this implies a specific word size. In this mode, the source and destination are specified down to the byte address. The source data is read in reverse order and written to the destination in increasing order. Transfers can be memory to memory or memory to memory mapped I/O. Figure 12-6 illustrates a memory to memory data transfer between unaligned 64-bit, source and destination addresses when the source is in decrement mode and the destination is in increment mode.

Figure 12-6. Source in Decrement and Destination in Increment Mode Transfer (Byte Reversal)



12.5.2.3 Constant Address Modes

In constant address mode, there is built-in support for “mailbox” destinations in the memory mapped I/O subsystem. A mailbox is a single or limited set of addresses used to collect information for dispersal later to their actual destination addresses by the receiving device. In constant address mode, one, two, or four bytes will be sent repeatedly until the byte count is satisfied.

The source address can be byte aligned; however, unlike other transfer modes, in constant address Mode the destination address must be aligned to the granularity size. No errors will be flagged if the destination address is not matched to the granularity, but the required lower address bits will be ignored. Additionally, software must ensure that the transfer byte count is an integer multiple of the granularity size. No error will be flagged if the transfer byte count is not an integer multiple and the remaining bytes in the requested granularity will be padded and transferred.

In this mode, the EDMA will write to the address that is placed in the DAR/DUAR pair with the transfer size specified in the granularity field. The transfers will continue until the byte count register is satisfied. Logically, transfers will only be memory to memory mapped I/O.

Constant address mode can be used with an increasing or decreasing source address to present data to the external device in either order.

Figure 12-7 through Figure 12-9 illustrate data transfers between an unaligned 64-bit, source and a constant destination addresses when the source is in increment mode.

Figure 12-7. Source in Increment and Destination in 1-Byte Granularity Constant Mode Transfer

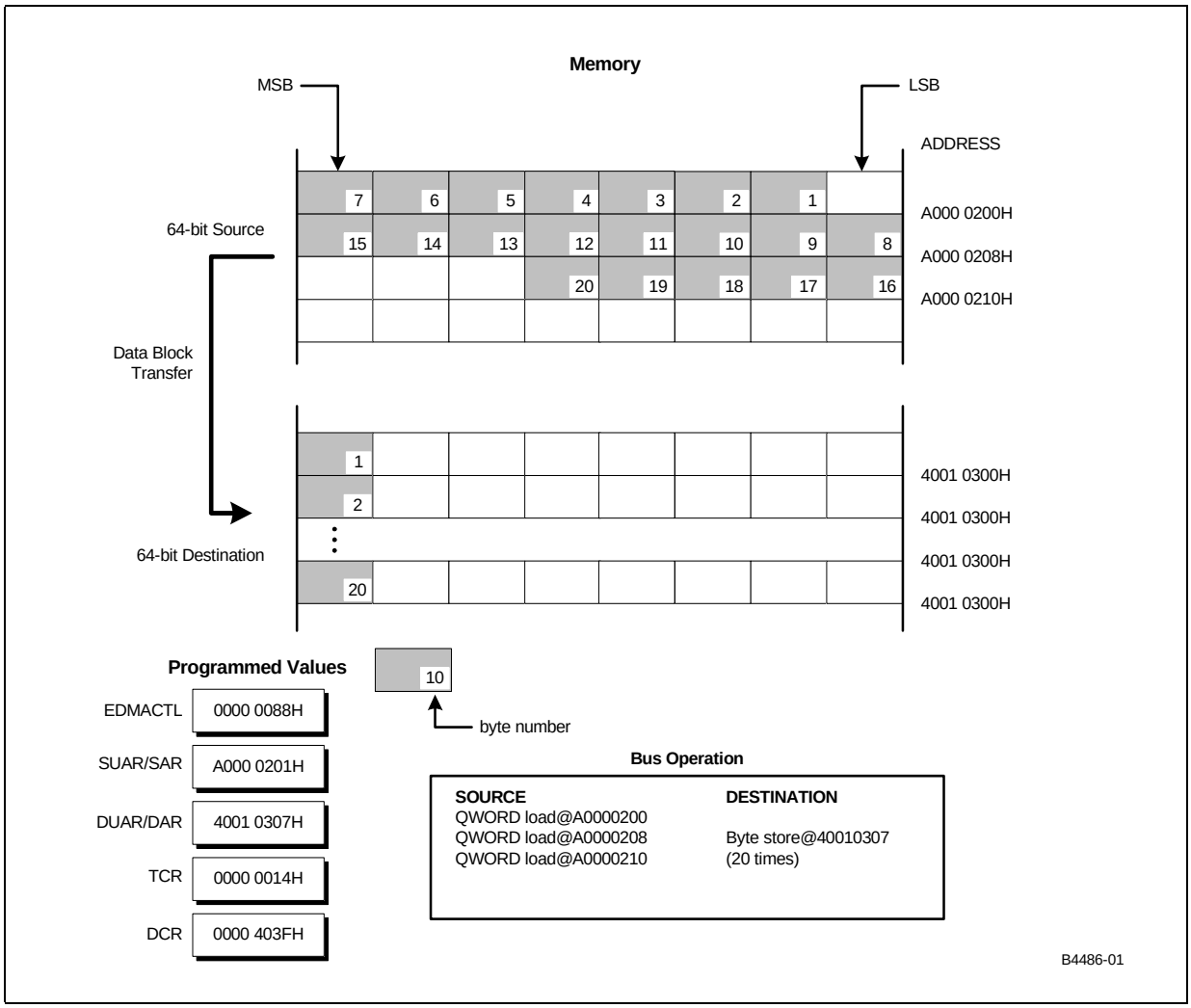


Figure 12-8. Source in Increment and Destination in 2-Byte Granularity Constant Mode Transfer

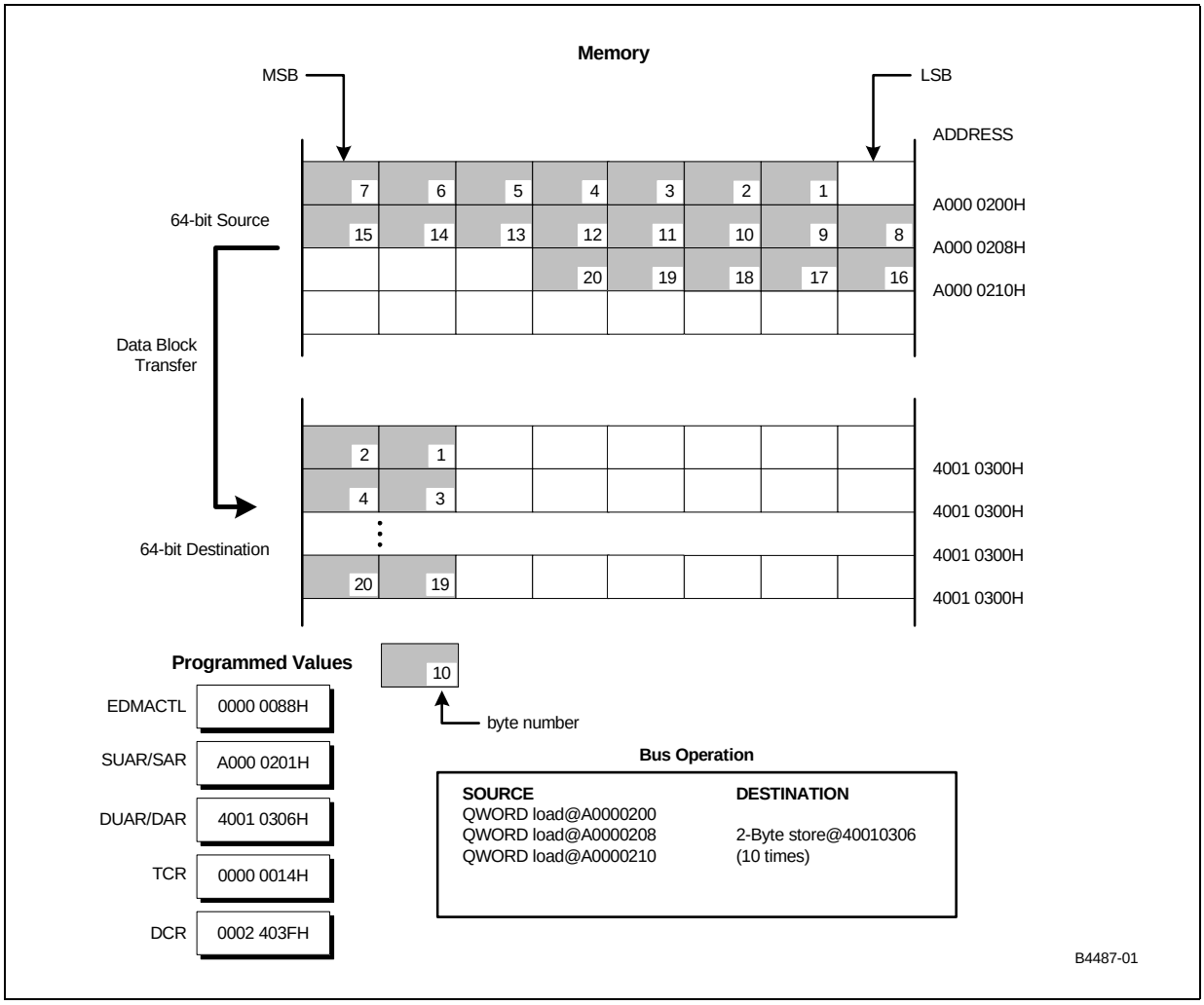


Figure 12-9. Source in Increment and Destination in 4-Byte Granularity Constant Mode Transfer

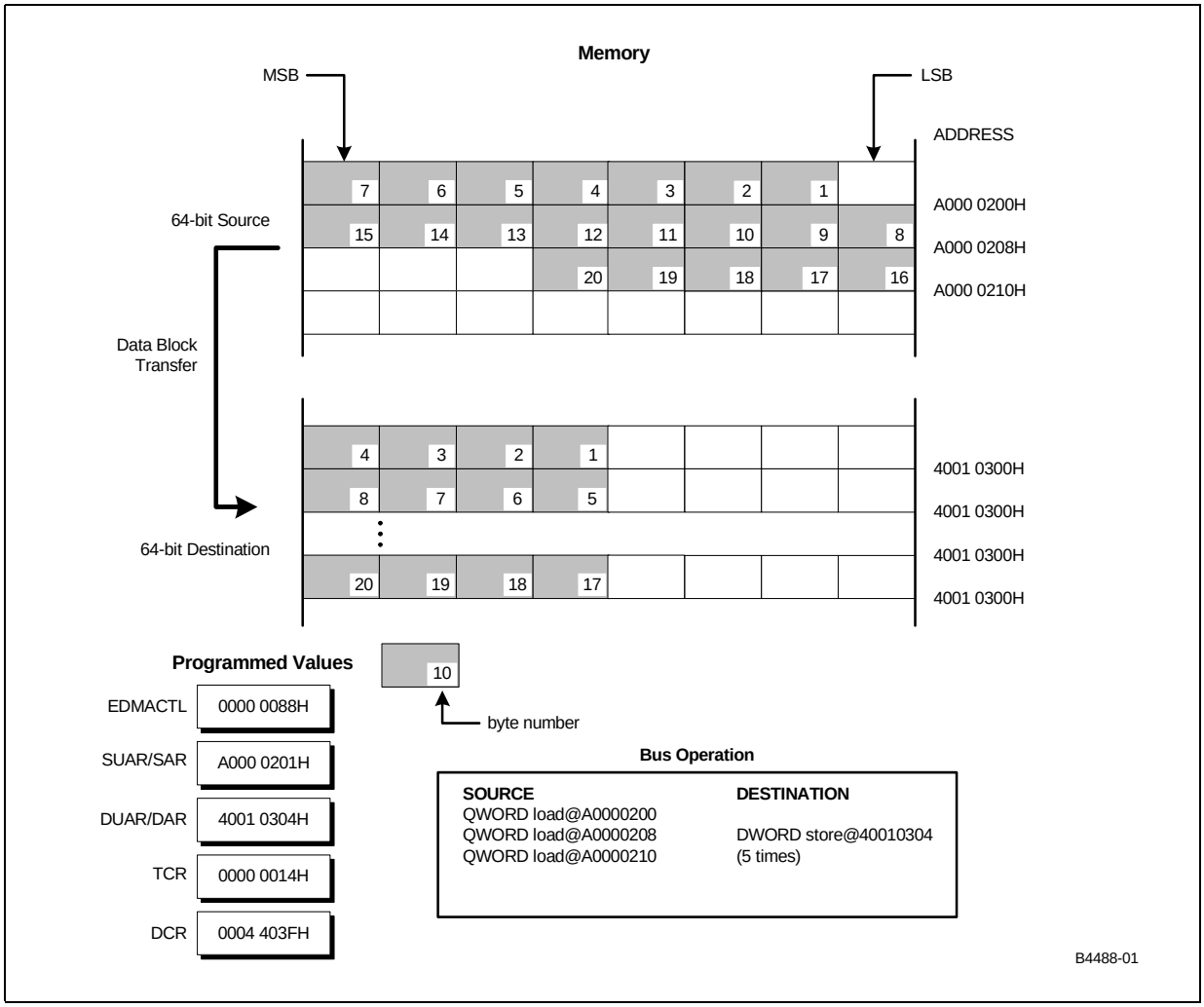


Figure 12-10, Figure 12-11, and Figure 12-12 illustrate data transfers between an unaligned 64-bit, source and a constant destination addresses when the source is in decrement mode.

Figure 12-10.Source in Decrement and Destination in 1-Byte Granularity Constant Mode Transfer

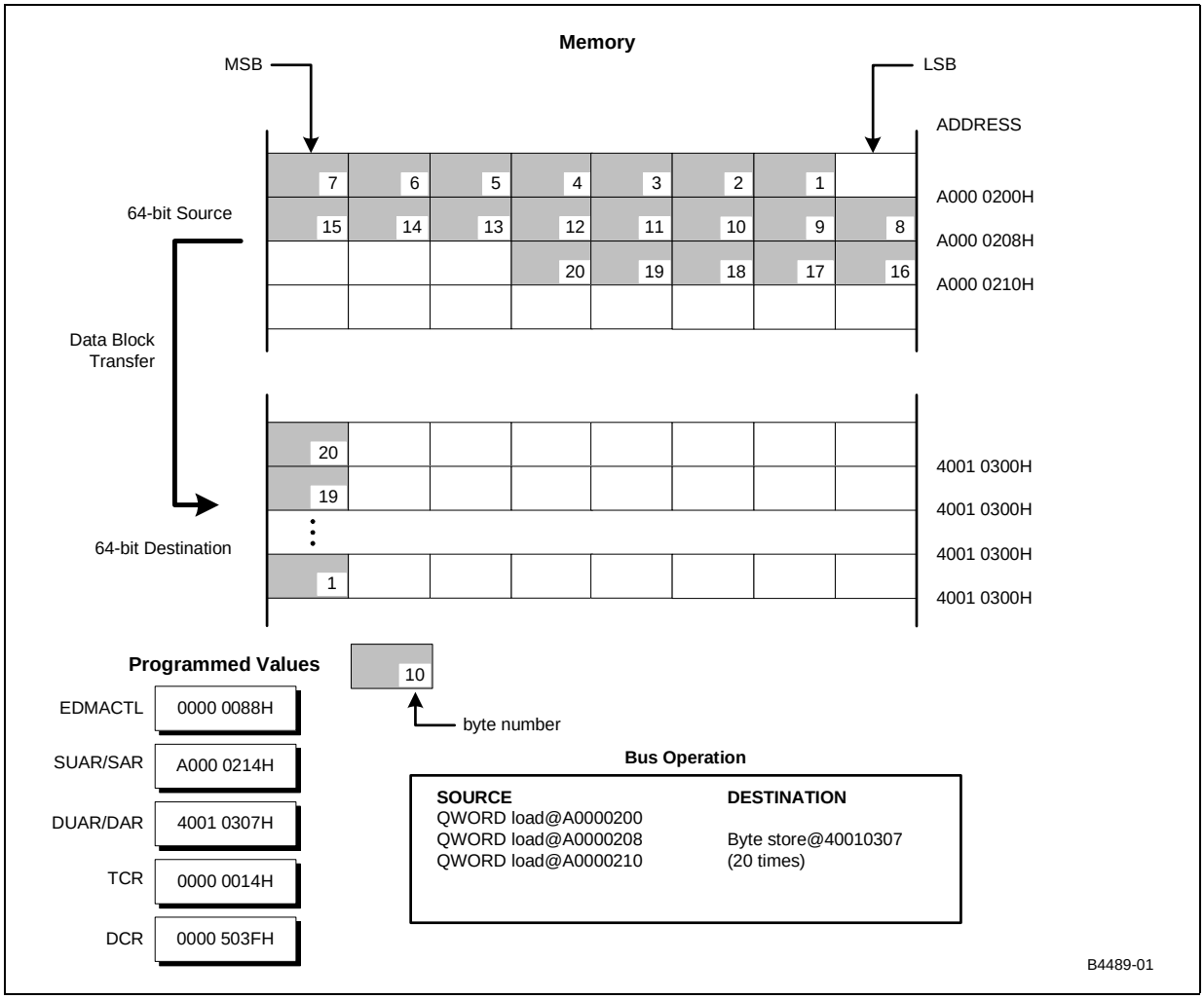


Figure 12-11.Source in Decrement and Destination in 2-Byte Granularity Constant Mode Transfer

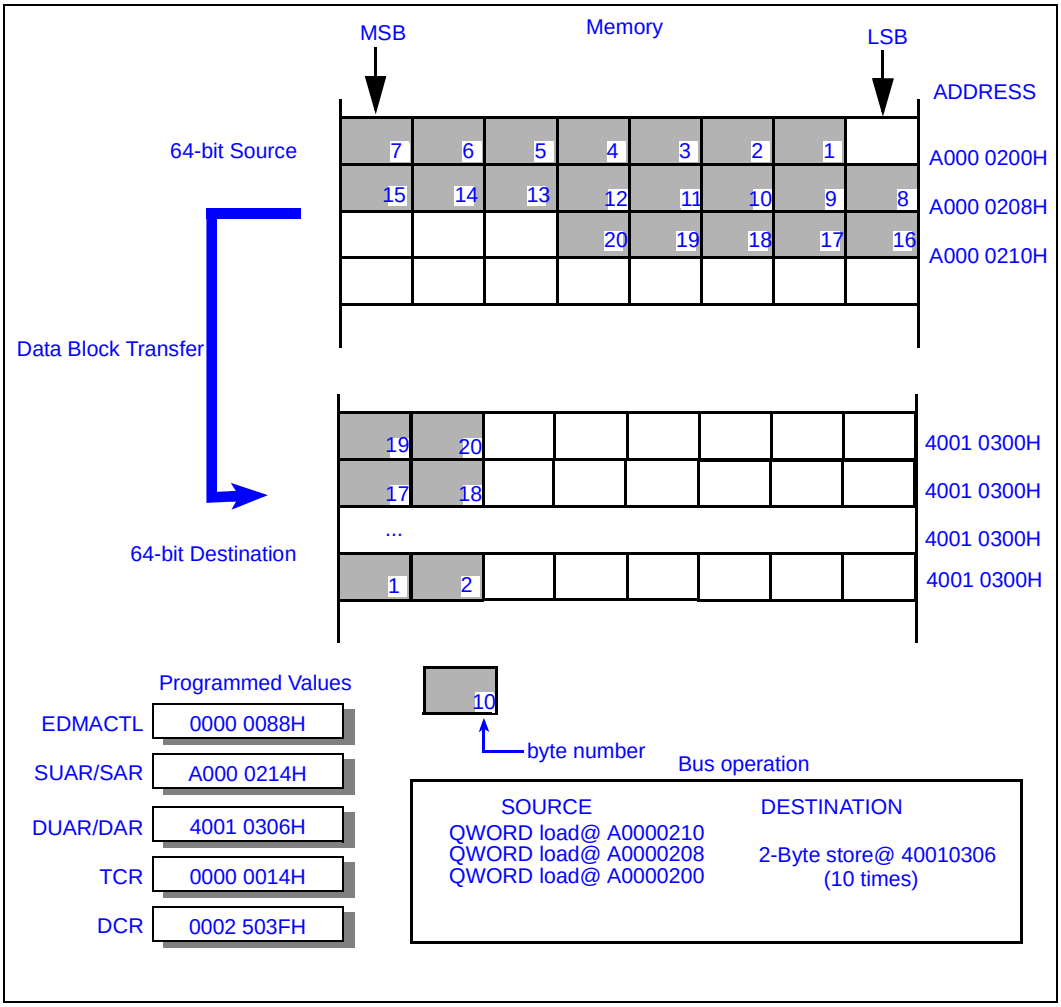
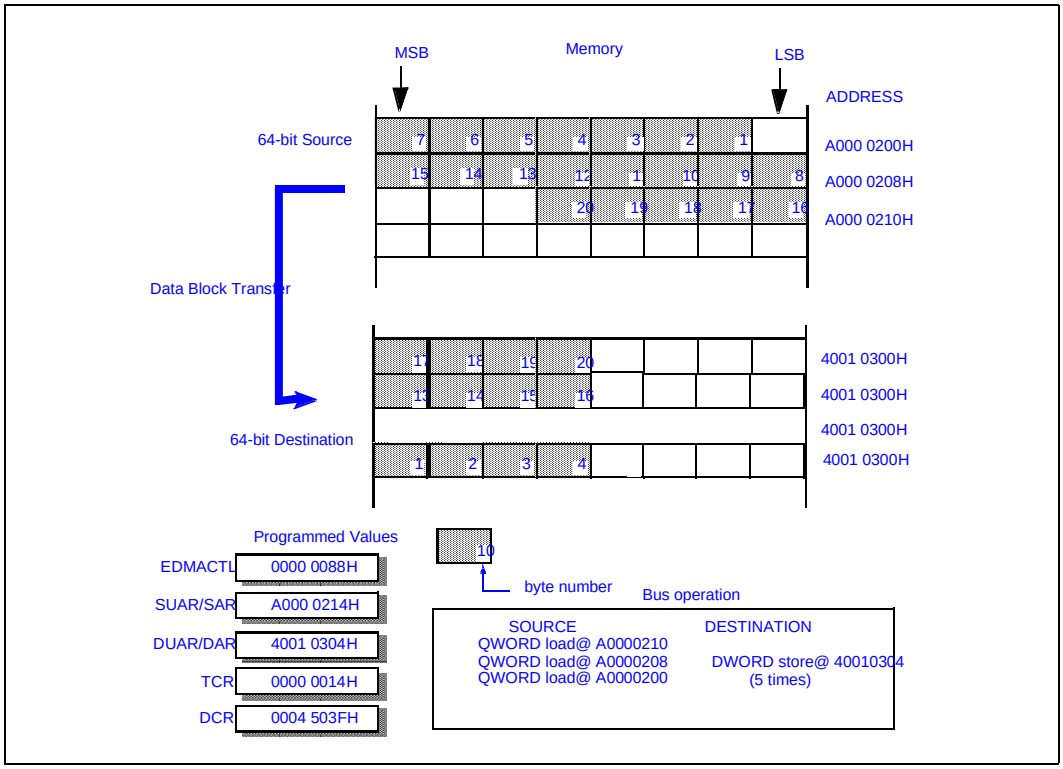


Figure 12-12.Source in Decrement and Destination in 4-Byte Granularity Constant Mode Transfer



12.5.2.4 Buffer and Memory Initialization Modes

The EDMA can be used to write a constant value to local memory or to memory mapped I/O. As with normal transfers, descriptors are used to specify the memory blocks to which the data contained in the Source Address Register is written.

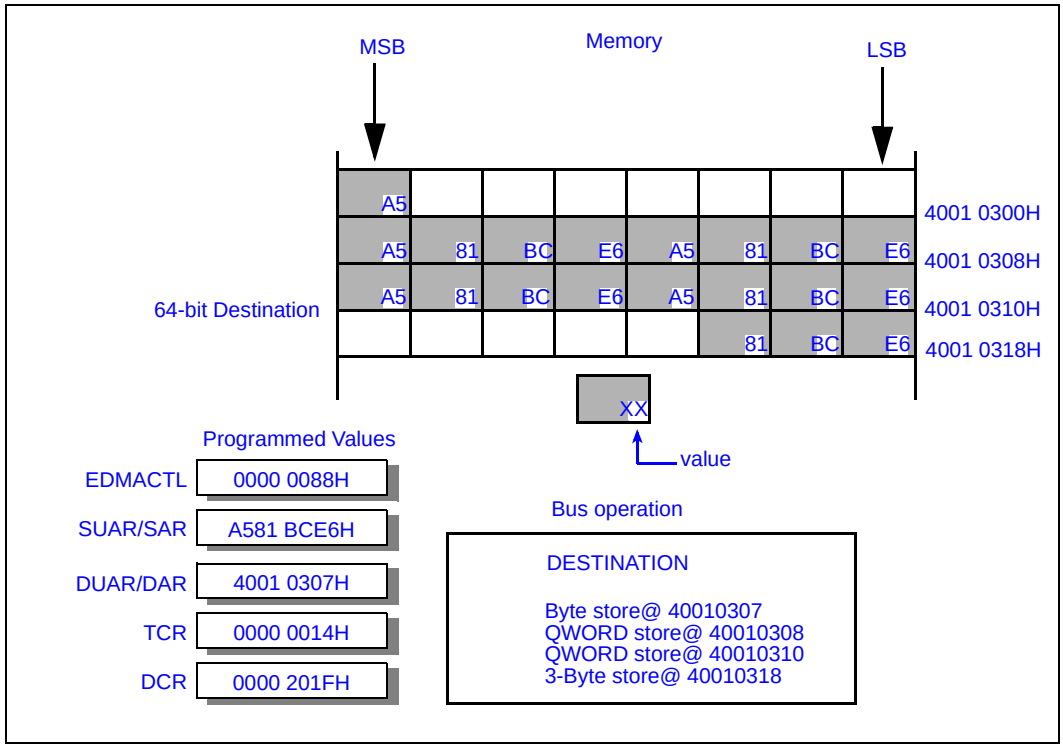
When buffer or memory initialization modes are selected, the data in the SAR is sent to the destination address. No data is fetched. Data is transferred in 32-bit replicated chunks to the destination. The transfers will continue until the byte count register is satisfied.

12.5.2.4.1 Memory Initialization Mode

Memory initialization mode transfers can be to memory or to memory mapped I/O. In this mode, the destination can be specified down to the byte address.

Figure 12-13 illustrates memory initialization to an arbitrary destination address.

Figure 12-13.Source in Memory Initialization and Destination in Increment Mode Transfer



12.5.2.4.2 Buffer Initialization Mode

Buffer initialization mode transfers will logically only be to memory mapped I/O and will utilize the constant destination mode and granularity fields of the DCR. The address granularity is dictated by the granularity field in the DCR. No errors will be flagged if the destination address is not matched to the granularity, but the required lower address bits will be ignored.

Figure 12-14 though Figure 12-16 illustrate buffer initialization mode to an arbitrary destination address.

Figure 12-14.Source in Buffer Initialization and Destination in 1-Byte Granularity Constant Mode Transfer

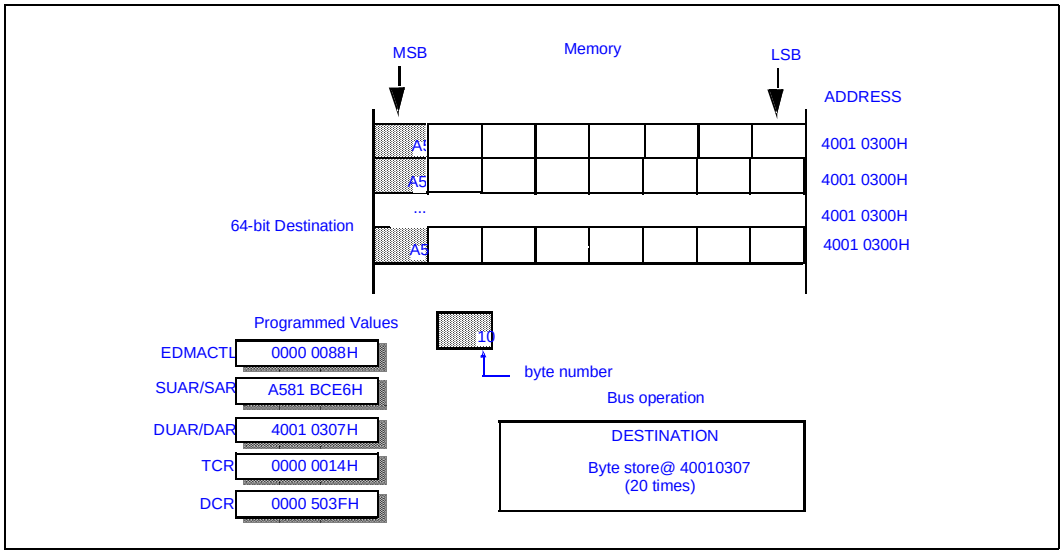


Figure 12-15.Source in Buffer Initialization and Destination in 2-Byte Granularity Constant Mode Transfer

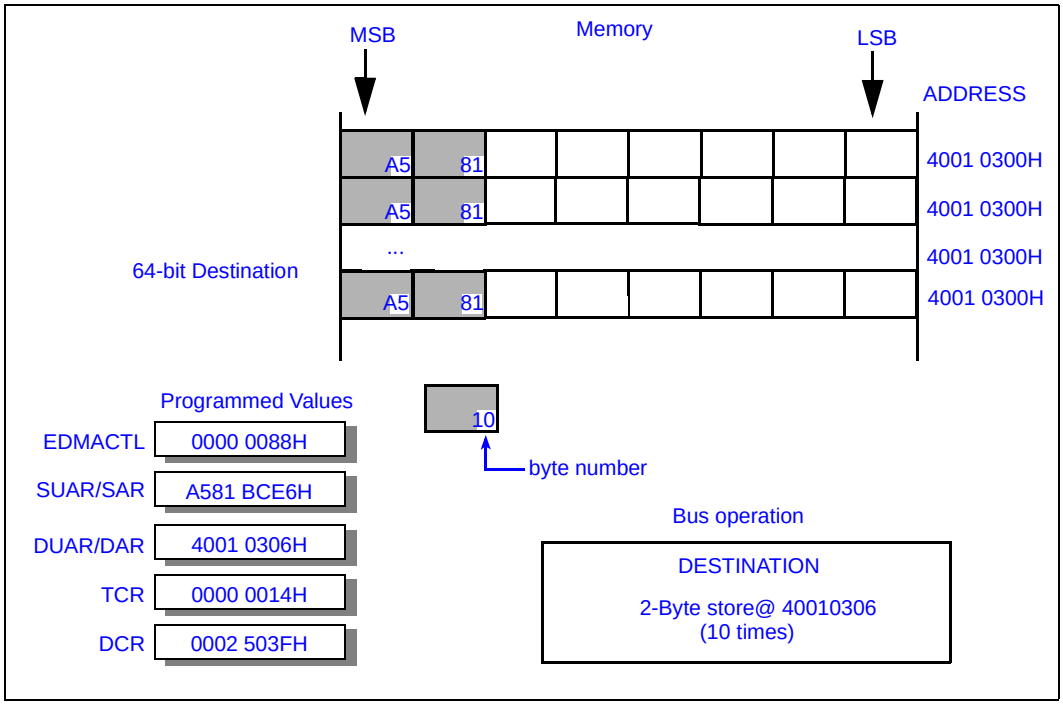
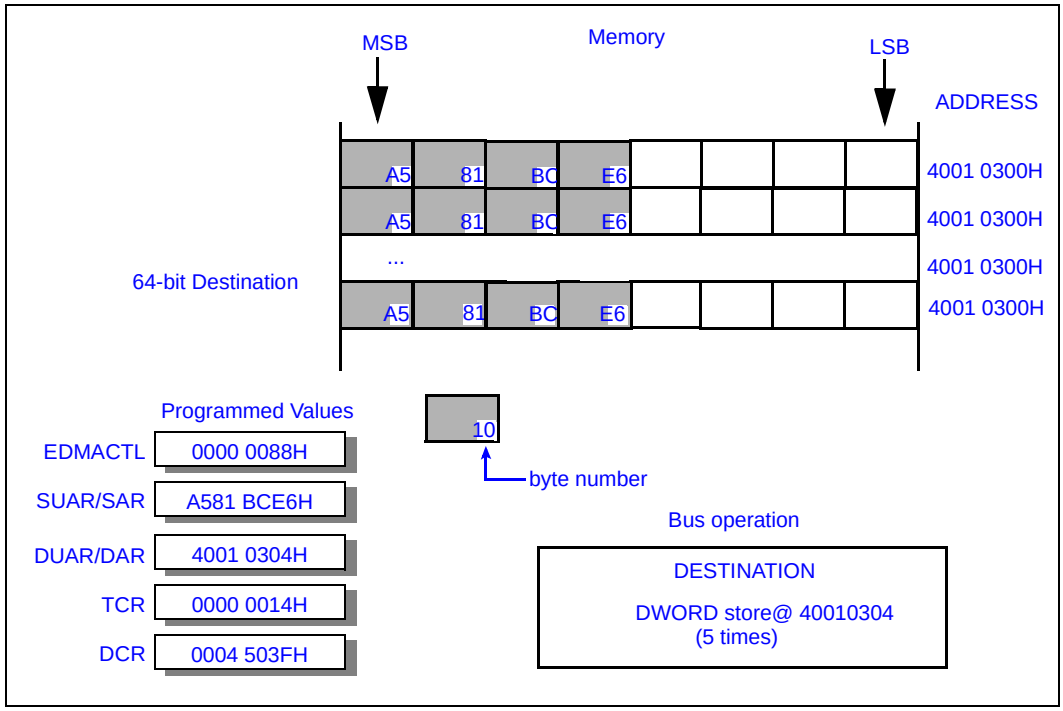


Figure 12-16.Source in Buffer Initialization and Destination in 4-Byte Granularity Constant Mode Transfer



12.5.3 PCI Express Traffic Class

To provide traffic shaping and quality of service within the system fabric, the EDMA contains the traffic class field within the DCR. Traffic class is provided to the external switches, bridges, and other fabric devices to allow one transaction to pass another based on a software generated priority map. The three bit field in the DCR is directly copied to the transaction traffic class field in the PCI-Express header as write transactions are generated to PCI-Express. This field is ignored for all other destinations.

12.6 Channel Data Queuing

Each channel contains a data buffer that is four cache-lines in size (256 bytes). The data buffer holds data temporarily to facilitate pipelining and hide latency, improving the throughput of data transfers between the source and destination.

12.7 Error Conditions

Any of several possible error conditions may arise during a transfer depending on which interfaces the transfer utilizes. The interfaces covered are the EDMA controller interface, the memory interface, and the I/O subsystem destination port interface. All error conditions are reported by setting the corresponding error bits in the Channel Status Register (CSR).

The subsections below describe all possible errors at each interface that the EDMA controller must detect and report. For those errors resulting in a channel abort, the response to the error is highly configurable. The controller may be configured to

generate a IA-32 core interrupt upon detection of an error. Beyond this mechanism, errors detected and logged in the CSR may be escalated as described in the chapter on RASUM and exception handling.

Refer to the EDMA access disposition tables in Chapter 10.0, "System Address Map," for an overview of defined ranges in the memory map and associated EDMA access treatment.

12.7.1 Controller Interface Error

The following errors may be reported for any EDMA initiated access, regardless of target interface:

- Illegal NDAR address
  - The descriptor pointer in NDAR is not naturally eight Dword aligned
  - The value in NDAR does not point to a valid memory location
- Illegal source address
  - Address does not comply with the Source Type bit in the DCR
  - Address out of range
- Illegal destination address
  - Address does not comply with the Destination Type bit in the DCR
  - Address out of range
  - Accesses to any IICH address region via the NSI
- Data parity error (corrupt data) returned by memory read retrieving descriptor information

All controller interface errors are fatal to the transfer in process and will result in channel abort.

Note: This includes data parity errors, since an error in reading a descriptor implies a corrupt descriptor in main memory, and in this case, it is impossible for the channel to determine precisely what part of the descriptor is damaged. The automatic abort upon detection of a corrupt descriptor is necessary to prevent any further data corruption as a result of its execution.

12.7.2 Memory Interface Error

The following errors may be reported for a EDMA initiated access (read or write) on the local memory interface:

- Addressing error (source or destination)
  - Physical EDMA address above REMAPLIMIT (see Section 10.6.1)
  - Physical address not allocated to memory (including PAM destination mapping)
  - Physical address specified an illegal memory destination (e.g., protected SMM range)
- Data parity error in reading data from the EDMA Data Queue
- Data parity error (poisoned data) returned by memory read for payload data

There is no time-out mechanism associated with transfers. The EDMA channel assumes that all reads to memory will eventually return, although they may return corrupt data, and will wait indefinitely for an outstanding read.

Addressing errors are fatal and will result in a EDMA channel abort, logged in CSR bit 4. The channel response to data errors is configurable; the channel may be programmed to abort, or to propagate the bad data to its destination.

12.7.3 I/O Interface Error

The following errors may be reported for a EDMA initiated access (write) on an I/O interface:

- Address crossed into a memory destination range (checked on each write access)
- Address crossed to a new destination port during a transfer (checked at 4 KB boundaries)

The latter error will result if poorly formed destination descriptor information specifies a length plus address combination that crosses the addressing boundary between independent outbound ports on the I/O subsystem. Any transfer with a destination range crossing an aligned 4 KB boundary in address space may encounter this error.

Addressing errors are fatal and will result in a channel abort. Channel response to data errors is configurable; each channel may be programmed to abort, or to propagate the corrupt data to its destination.

12.8 Channel Arbitration

Arbitration among the four independent channels occurs in two stages. Each channel has an independent bus request/grant pair to the arbiter internal to the controller. The controller in turn has a single request/grant pair to the main arbiter. The arbiter within the controller handles the fairness among channels, while the inbound/outbound arbiter handles fairness between the EDMA channels and other competing traffic sources.

The internal arbiter uses a strict round-robin policy, with the added modification of an optional “high priority” designation for one channel at any given time. Thus a set of competing channels will achieve balanced bandwidth performance during normal operation.

The inbound/outbound arbiter provides a programmable single or double “grant duration” for the EDMA controller. Thus the channel that “wins” internal arbitration may be allowed to issue one or two access requests back-to-back in a single arbitration cycle. The second request is accepted if the inbound/outbound arbiter is programmed to a grant count of 2 the requesting channel has two consecutive requests of the same type targeting the same destination ready to send, and there are sufficient command and data resources available for the second request.

12.8.1 Normal Arbitration Scheme

A fully connected round-robin arbiter provides a distinctive balanced service among competing requestors. Each of the actively competing channels will receive an equal fraction of the bandwidth service provided by the inbound/outbound arbiter on behalf of each EDMA channel. In the absence of any competition from the IA-32 core, PCI Express ports, or other I/O, each EDMA channel will be allowed to saturate the memory interface. For example, given a memory interface “saturation point” of 4 GB/s, the round-robin scheme would be equally distributed between the competing EDMA channels.

12.8.2 Prioritized Arbitration Scheme

The high priority option in the arbitration scheme provides for a single “high priority” channel to receive favorable latency and bandwidth service in the face of multiple competing “normal priority” channels. This is accomplished by designating the high priority channel using a priority enable bit and a two-bit field (to select one of four channels) in the EDMA control register.

The internal EDMA controller arbiter modifies its arbitration algorithm to provide a grant to the priority channel between the grants for each of the other channels, which retain their round-robin prioritization relative to each other. For example, given a hypothetical memory interface bandwidth of 4 GB/s, the priority-modified scheme would result in 2 GB/s (half the available bandwidth) for the designated priority channel, and the remaining 2 GB/s split equally among the other competing channels.

The limitation that a single channel at a time be designated as the priority channel is an acknowledgement that quality of service differences, given multiple “priority” channels, would be slight in this implementation with two-level arbitration and only four competing channels. It is anticipated that software will be able to determine when an application is particularly sensitive to service level, has been allocated a channel, and will manage the assignment of priority accordingly. If more than one such “sensitive” application is in flight at the same time, it is perceived to be more efficacious to allow fair competition between those sources, and let the kernel or device driver software attempt to manage competition for resources at the system level to prevent service level problems.

Each EDMA channel supports dynamic modification to the priority channel settings (while one or more channels are active). A write to the control register that changes the priority channel configuration takes affect at the next arbitration decision point after the write has completed. There is no direct interlock between the arbiter configuration and any of the active channels. Such an event is effectively an environment change, orthogonal to work in progress on any given channel.

12.9 Configuration

The EDMA controller uses memory-mapped configuration registers for the majority of its per channel register sets. The controller is software compatible with standard PCI device configuration and implements a standard PCI header in its configuration-mapped register set as shown in Figure 12-1. The memory-mapped register space associated with the controller is identified by a 32-bit memory Base Address Register (BAR). Table 12-1 provides an overview of the memory-mapped register set for a representative channel of the controller.

Table 12-1. Channel 0 Memory-mapped Register Set

Memory Mapped I/O for EDMA Channel 0	Memory offset	Access	Size	Default	Sticky
Channel Control Register (CCR0)	00-3h	RW	32 bits	0000_0000h	No
Channel Status Register (CSR0)	04-07h	RWC, RO	32 bits	0000_0000h	No
Current Descriptor Addr Reg (CDAR0)	08-0Bh	RO	32 bits	0000_0000h	No
Current Descriptor Upper Addr Reg (CDUAR0)	0C-0Fh	RO	32 bits	0000_0000h	No
Source Address Register (SAR0)	10-13h	RO	32 bits	0000_0000h	No
Source Upper Address Register (SUAR0)	14-17h	RO	32 bits	0000_0000h	No
Destination Address Register (DAR0)	18-1Bh	RO	32 bits	0000_0000h	No
Destination Upper Address Register (DUAR0)	1C-1Fh	RO	32 bits	0000_0000h	No

Table 12-1. Channel 0 Memory-mapped Register Set

Next Descriptor Address Register (NDAR0)	20-23h	RWL	32 bits	0000_0000h	No
Next Descriptor Upper Address Register (NDUAR0)	24-27h	RWL	32 bits	0000_0000h	No
Transfer Count Register (TCR0)	28-2Bh	RO	32 bits	0000_0000h	No
Descriptor Control Register (DCR0)	2C-2Fh	RO	32 bits	0000_0000h	No

All internal registers are accessible through host-initiated configuration space accesses or SMBus interface accesses. Internal registers are not accessible from the I/O subsystem interfaces.

12.9.1 Power Up/Default Status

Upon power-up or hardware reset, the channel registers are initialized to their default values. All reserved and unimplemented registers and bits in the device return zero on reads and are unaffected by writes.

12.9.2 Channel-Specific Register Definitions

Each channel has twelve 32-bit memory-mapped registers for its independent operation. Eight of these registers (refer to the descriptions below) are loaded automatically from their corresponding fields in the chain descriptor when a new descriptor is fetched from local memory during normal operation. The format of the corresponding descriptor fields in memory is identical to the format defined for the channel-specific registers. Refer to "Memory Mapped I/O for EDMA Registers" on page 651 for bit definitions.

Read/write access is available only to the following:

- Channel Control Register (CCR)
- Channel Status Register (CSR)
- Next Descriptor Address Register (NDAR)
- Next Descriptor Upper Address Register (NDUAR)

The remaining registers are read-only and are automatically loaded with new values defined by the chain descriptor whenever the channel reads a chain descriptor from local system memory.

Note: Automatic loading of the channel-specific registers occurs after the memory read completion returns the descriptor data (32 B), and verification has taken place. Verification includes checking parity on the data returned and checking that the channel is properly configured to receive new descriptor data. (If a suspend is in progress, the descriptor data will be dropped in honor of the suspend.)

12.9.2.1 Channel Control Register – CCR

The Channel Control Register (CCR) specifies the overall operating environment for the channel. This is a read/write register, and is cleared to zero on power-on or system reset (contains no sticky bits). Application software initializes this register only after initializing the chain descriptors in system memory and updating the Next Address Registers with the location of the first chain descriptor in memory. The CCR may be written when the channel is active to modify channel operation (stop, suspend, etc.) while the channel is active.

The following bits are defined in the CCR:

- Start: initiate a new transfer (requires that the CSR be appropriately cleared)
- Stop: abort the current transfer (immediately)
- Suspend: suspend the current transfer (upon completion of the current descriptor)
- Channel Resume: resume a suspended transfer (retrieve the descriptor indicated by NDAR/NDUAR from local memory, and proceed with execution per the value returned). Requires that the stopped and abort status bits in the CSR be clear to take effect and will automatically clear end of chain and end of transfer flags.

Refer to "Offset 00h: CCR0 - Channel 0 Channel Control Register" on page 653 for the format of the CCR.

12.9.2.2 Channel Status Register – CSR

Channel Status Register (CSR) contains flags to indicate the channel status. The register is read by application software to get the current channel status and to determine the source of interrupts. CSR is cleared to zero on power-on or system reset. This is a read/write register.

The following bits are defined in CSR:

- Channel Active: transfer in progress
- Aborted: transfer encountered an error
- Stopped: transfer stopped via software request (Stop bit detected)
- Suspended: transfer suspended via software request (Suspend bit detected)
- End of Transfer: channel has completed execution of (at least one) descriptor
- End of Chain: channel has completed execution of the terminal descriptor (null NDAR/NDUAR)

Refer to "Offset 04h: CSR0 - Channel 0 Channel Status Register" on page 656 for the format of CSR.

12.9.2.3 Current Descriptor Address Register – CDAR

The Current Descriptor Address Register (CDAR) contains the lower 32-bits of the address for the current chain descriptor in local system memory. The CDAR is cleared to zero on power-on or system reset, and is loaded automatically with the value from the Next Descriptor Address Register (NDAR) when a new block transfer is initiated. This register is read-only, and may be polled by software to monitor the progress of the channel as it traverses the descriptor chain.[1]

12.9.2.4 Current Descriptor Upper Address Register – CDUAR

The upper address will not be used in the EP80579, which is limited to 32-bit addressing.

The Current Descriptor Upper Address Register (CDUAR) contains the upper 32-bits of the address of the current chain descriptor in local system memory. The CDUAR is cleared to zero on power-on or system reset and is loaded automatically with the value from the Next Descriptor Upper Address Register (NDUAR) when a new block transfer is initiated. This register is read-only.[1]

1. Note that the IMCH does not provide an interlock to guarantee that consecutive reads to the CDAR/CDUAR pair return portions of the same descriptor in the event of a collision between the read accesses and a descriptor load operation. If software requires knowledge of the current descriptor, the "Suspend" function must be invoked prior to polling these registers.

12.9.2.5 Source Address Register – SAR

The Source Address Register (SAR) contains the lower 32-bits of the source address for the current transfer. The SAR is cleared to zero on power-on or system reset and is loaded automatically with the Source Address field of the chain descriptor (first DWord) when a new chain descriptor is read from memory. The address can be aligned to any byte boundary. The system destination for reads to this address range must match the Source Type setting of the DCR or the transfer will abort.

12.9.2.6 Source Upper Address Register – SUAR

The upper address will not be used in the EP80579, which is limited to 32-bit addressing.

The Source Upper Address Register (SUAR) contains the upper 32-bits of the source address for the current transfer. The SUAR is cleared to zero on power-on or system reset and is loaded automatically with the Source Upper Address field of the chain descriptor (second DWord) when a new chain descriptor is read from memory.

12.9.2.7 Destination Address Register – DAR

The Destination Address Register (DAR) contains the lower 32-bits of the destination address for the current transfer. The DAR is cleared to zero on power-on or system reset and is loaded automatically with the Destination Address field of the chain descriptor (third DWord) when a new chain descriptor is read from memory. The address can be aligned to any byte boundary. The system destination for writes to this address range must match the Destination Type setting of the DCR or the transfer will abort.

12.9.2.8 Destination Upper Address Register – DUAR

The upper address will not be used in the EP80579, which is limited to 32-bit addressing.

The Destination Upper Address Register (DUAR) contains the upper 32-bits of the destination address for the current transfer. The DUAR is cleared to zero on power-on or system reset and is loaded automatically with the Destination Upper Address field of the chain descriptor (fourth DWord) when a new chain descriptor is read from memory.

12.9.2.9 Next Descriptor Address Register – NDAR

The Next Descriptor Address Register (NDAR) contains the lower 32-bit address of the next descriptor chain in the local system memory. The NDAR is cleared to zero on power-on or system reset and is loaded automatically with the Next Descriptor Address field of the chain descriptor (fifth DWord) when a new chain descriptor is read from memory. This address must be aligned to an 8-DWord address boundary. A value of zero implies the end of chain if the value of Next Descriptor Upper Address (loaded into the NDUAR) is also zero. Application software writes this register with the address of the first chain descriptor in memory prior to initiating a transfer.

Note: The application software must make sure that the Start bit in the CCR and the Channel Active bit in the CSR are clear prior to writing to the NDAR. The IMCH protects this register from being written when these bits are not clear. If the NDAR and NDUAR are zero when the Start bit is set, no transfer will be initiated.

12.9.2.10 Next Descriptor Upper Address Register – NDUAR

The upper address will not be used in the EP80579, which is limited to 32-bit addressing.

The Next Descriptor Upper Address Register (NDUAR) contains the upper 32-bit address of the next descriptor chain in the local system memory. All address bits above bit 35 must be zero or the transfer will abort and an error will be reported. A value of zero implies the end of chain if the value of Next Descriptor Address (loaded into the NDAR) is also zero. NDUAR is cleared to zero on power-on or system reset and is loaded automatically with the Next Descriptor Upper Address field of the chain descriptor (sixth DWord) when a new chain descriptor is read from memory. Application software (likely the device driver) writes this register with the address of the first chain descriptor in the memory prior to initiating a transfer.

Note: The application software must make sure that the Start bit in the CCR and the Channel Active bit in the CSR are clear prior to writing to the NDUAR. The IMCH protects this register from being written when these bits are not clear. If the NDAR and NDUAR are zero when the Start bit is set, no transfer will be initiated.

12.9.2.11 Transfer Count Register – TCR

The Transfer Count Register (TCR) contains the length of the current transfer in bytes. The TCR is cleared to zero on power-on or system reset and is loaded automatically with the Transfer Count field of the chain descriptor (seventh DWord) when a new chain descriptor is read from memory. The TCR allows for a maximum transfer of 16 MB, commensurate with current operating system capabilities. A value of zero is valid and results in no data being transferred and no cycles generated on the source or destination buses. It also results in completion bits being set after successful completion “same as if it were a no zero length transfer”.

12.9.2.12 Descriptor Control Register – DCR

The Descriptor Control Register (DCR) contains control values for the transfer on a per descriptor basis. The DCR is cleared to zero on power-on or system reset and is loaded automatically with the Descriptor Control field of the chain descriptor (eighth DWord) when a new chain descriptor is read from memory. The values in the DCR may vary for different descriptors within a single chain.

Note: The descriptor control register value stipulates coherence attributes for both the source and destination addresses defined by this chain descriptor. Independent bits are also defined to specify whether the source and destination address ranges are to be treated as “coherent” or “non-coherent” by the IMCH. When the DCR value stipulates that one or both of the source and destination are to be treated as “non-coherent” space, the IMCH will rely on software to maintain system memory coherency and will not issue FSB cycles during the block transfer to snoop processor caches on behalf of the corresponding address range(s).

The following bits are defined in the CCR:

- Destination Address Mode: two bits specify destination address as increment or constant
- Granularity of the transfer in destination constant address mode: two bits (1B, 2B, or 4B)
- PCI-Express Destination Traffic Class: three bits define this traffic class
- Source Address Mode: two bits specify source address as increment, decrement, or buffer/memory initialization
- Buffer/Memory Initialization Mode: Specifies a write to fill an area of memory

- Destination Coherency: specifies whether destination addresses should be snooped on the FSB
- Source Coherency: specifies whether source addresses should be snooped on the FSB
- Destination Type: specifies whether the destination is local memory or the I/O subsystem
- Source Type: specifies whether the source is local memory or the I/O subsystem (defined only for symmetry, I/O subsystem source addresses are not supported)
- Abort Interrupt Enable: specifies whether to generate an interrupt on abort
- Stop Interrupt Enable: specifies whether to generate an interrupt on stop
- Suspend Interrupt Enable: specifies whether to generate an interrupt on suspend
- End of Transfer Interrupt Enable: specifies whether to generate an interrupt on EOT
- End of Chain Interrupt Enable: specifies whether to generate an interrupt on EOC

Refer to Chapter , “Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register,” for the format of the DCR.

12.10 Interrupts

Each EDMA channel can be configured to generate interrupts to the processor interface. The interrupt enable bits for end of transfer and end of chain in the Descriptor Control Register (DCR) determine if the channel generates an interrupt upon successful error-free completion of a transfer. The Abort Interrupt Enable bit in the DCR determines if the channel generates an interrupt upon encountering an error. Refer to “Error Conditions” on page 329 for details on errors on both the source and destination interface. Table 12-2 summarizes the status flags, and the conditions under which interrupts will be generated.

Each chain descriptor can independently set or clear the various interrupt enable bits in the Descriptor Control Register. This level of control for interrupt generation permits flexibility in synchronization between application software and transfers in progress. If interrupts are not enabled, synchronization can be achieved by polling the status bits in the Channel Status Register (CSR).

Note: “-” In the table below equates to a non valid combination

Table 12-2. Interrupt Summary

Interrupt Conditions	Channel Status Register (CSR) Flags						DCR Bit Settings (INTR Enable Bits)				
	Channel Active	Stopped	Suspended	End of Transfer	End of Chain	Channel Aborted (Error)	Stop INTR Enable	Suspend INTR Enable	EOT INTR Enable	EOC INTR Enable	Abort INTR Enable
Stopped	0	1	0[2]	-	-	-	1[4]	-	-	-	-
Suspended	0	0[2]	1	1[3]	-	-	-	1	-	-	-
End of Transfer	1	-	-	1	-	-	-	-	1	-	-
End of Chain	0	-	-	1	1	-	-	-	-	1	-
Channel Abort[1]	0	-	-	0	0	1	-	-	-	-	1

Notes:

1. The IMCH ensures that any aborted transfer will be reported via the Channel Abort status bit and that this bit will never be accompanied by an End of Transfer or End of Chain indication. This ensures that software never mistakes an aborted transfer for a successfully completed transfer – even if the error is not detected until the final write to the final destination address of the terminal chain descriptor.
2. The Stop and Suspend functions are mutually exclusive, and only one of the two status bits will ever be asserted by the IMCH. In the event that software asserts both controls in the CCR, the Stop function will take precedence.
3. The EDMA Suspend function causes the channel to suspend operation at the completion of the current descriptor. The EOT status bit will always accompany the suspended status bit. Note that even if interrupts are enabled for both EOT and suspend, only a single interrupt event will result.
4. The Stop function causes the channel to abort the transfer in progress immediately. It is recommended that software read back the channel status register to verify that a stop command has taken effect, since this will be much faster than setting the interrupt enable for stop and waiting for the interrupt to occur.

12.10.1 Interrupt Routing Mechanisms

Two different mechanisms are available to route interrupts generated by channels to the IA-32 core. Note that the interrupt mechanism itself is not channel-specific; all channels generating interrupts share the same interrupt vector and handler. This is in line with the expectation that a single device driver controls each EDMA channel at large, rather than independent drivers per channel.

The first interrupt mechanism uses the integrated IOxAPIC and 8259 emulation hardware. All interrupts from the channels are logically OR'ed and routed to the interface controller for propagation via the in-band Assert_Intx and Deassert_Intx special cycles, emulating a level-sensitive interrupt output. The IMCH tracks these special cycles, and forwards the signaled interrupt to the IA-32 core. If the APIC enable bit is set, an interrupt will result in an APIC message. If the APIC enable bit is clear (unanticipated but possible), an interrupt will result in a legacy mode 8259-style level sensitive interrupt directly to the IA-32 core socket.

The second interrupt mechanism uses Message Signaled Interrupt (MSI) generation functionality integrated into the EDMA. Internal interrupt messaging utilizes the PCI message capability structure and does not support external interrupt input routing.

(That is, the limited MSI functionality described here is dedicated to the EDMA channel.) This second mechanism is preferred for interrupt signaling, but is only available in platforms running an MSI-capable operating system.

Selection between these two mechanisms is automatic in the IMCH. If MSI messaging is enabled, as indicated by the enable bit in the MSI control register, then the MSI interrupt mechanism is used. If MSI message generation is disabled, any initiated interrupt will use the dedicated pin legacy mechanism.

For both interrupt mechanisms, the interrupt service routine (ISR) must service all interrupts for all channels. The memory-mapped EDMA Controller Global Status Register must be statused before returning from the ISR to ensure no additional interrupts have occurred. Failure to address interrupts in all channels will result in potential system starvation.

12.10.2 Message Signaled Interrupt (MSI)

The EDMA controller is capable of generating upstream interrupt messages (MSI) directly to the IA-32 core. An MSI is signaled via a Memory Write to address 0FEEx_xxxxh.

Three 32-bit registers are required in the controller to support this mechanism. The default values of these registers are compatible with the default value of the IOxAPIC specification. The three registers are the MSI Control Register (MSICR), MSI Address Register (MSIAR), and MSI Data Register (MSIDR). Software must program these registers to appropriate values prior to enabling internal MSI functionality.

Note: It is unsafe to enable the integrated MSI APIC function of the controller in environments under control of a non MSI-capable operating system.

The MSI mechanism supports differentiation between interrupts generated during normal operation (EOT, EOC, stop, and suspend) and interrupts due to errors (abort). This extra level of granularity is unavailable via the legacy interrupt mechanism.

To facilitate use of a single device driver for the entire EDMA function, a single MSI register set services all channels. The support for two different messages on behalf of the controller is included in the MSI register set. Refer to "EDMA Registers: Bus 0, Device 1, Function 0" on page 501 for the format of these registers.

Note: The integrated APIC functionality will not support level-sensitive interrupt emulation requiring the use of broadcast EOI cycles from the FSB. No path is provided to handle such traffic from IA-32 core to EDMA control engine. Thus the only supported MSI type is the edge-triggered variety.

The following subsections describe the register set for MSI support.

12.10.2.1 MSI Control Register – MSICR

The MSI Control Register (MSICR) contains control information for MSI interrupt capability. The multiple-message enable field and MSI enable are contained in this register.

12.10.2.2 MSI Address Register – MSIAR

The MSI Address Register (MSIAR) contains address information specifying the message destination address for MSI interrupts.

12.10.2.3 MSI Data Register – MSIDR

The MSI Data Register (MSIDR) contains routing and priority data for generation of MSI interrupts.

12.10.3 Interrupt Ordering

To support MSI signaling as transfers complete, the IMCH must take special steps in hardware to ensure that IA-32 core accesses to memory in response to MSI do not experience producer/consumer ordering failures. Specifically, the chip must internally guarantee functionality equivalent to a logical “FENCE” operation between the MSI and subsequent IA-32 core traffic from the FSB.

12.10.3.1 Interrupt Ordering for Memory Destination

The failure to be prevented for interrupts signaled at the end of transfers to memory destinations is as follows:

- Each EDMA channel is programmed to move a single cache-line of data and issue an MSI upon completion. This is a special case, for illustrative purposes; a similar scenario arises for the last few writes of a multi-line transfer.
- As soon as the write data is posted into the inbound/outbound arbiter headed for the memory interface, the MSI is issued directly to the FSB. Note that the transfer to memory may be issued without an accompanying FSB snoop cycle (non-coherent), thus the data and interrupt message are logically traversing independent traffic paths.
- In response to the MSI, the IA-32 core issues a memory read to retrieve the data from memory. In the absence of an internal interlock, this read may proceed to the memory controller before the posted write data is accepted into the memory controller from the inbound/outbound arbiter. This is where the error occurs, because the memory controller will not have any information regarding the relative issue order of the write and the read – if the read gets there first, it will retrieve stale data.

To prevent such a failure, the inbound/outbound arbiter includes specialized hardware to guarantee that all posted write data received ahead of an MSI are forwarded out of the arbiter before the MSI message will be forwarded to the FSB.

A similar interlock in the inbound/outbound arbiter prevents failures in the non-interrupt case. When software is polling the Channel Status Register (CSR) to detect transfer completion, specialized hardware guarantees that the read completion stalls until all prior posted write data is forwarded out of the arbiter.

12.10.3.2 Interrupt Ordering for Outbound Destination

The failure to be prevented for interrupts signaled at the end of transfers to outbound port destinations is as follows:

- Each EDMA channel is programmed for a single block transfer to an I/O device, with interrupt notification enabled at the completion of that transfer. For this example, MSI generation is disabled.
- When the final write of the transfer is posted into the inbound/outbound arbiter, a level-sensitive interrupt is signaled directly to the interrupt output pin, bypassing all internal queue structures.
- If the integrated IOxAPIC is enabled, the response to the interrupt pin will be an APIC message received. If the APIC is disabled, a sideband interrupt is signaled directly to the IA-32 core via a level sensitive output.

- For either case, the likely software response to the interrupt will be a posted write “door-bell” access to a memory mapped control register in the EDMA destination device to communicate completion of the transfer. (Semantic: EDMA transfer completed without errors, you have all the data, GO.)
- In the absence of internal interlocks as described above, multiple failures are possible. The APIC message could bypass EDMA data pending within the inbound/outbound arbiter, and the IA-32 core doorbell write could do the same. This would result in stale data “executed” in response to the doorbell.

The internal IMCH hardware interlock prevents APIC messages from being forwarded to the FSB while posted data remains pending in the inbound/outbound arbiter. This prevents most of the problematic behavior in this case. If the APIC is disabled, the IA-32 core must retrieve the interrupt vector from the 8259 emulator, and the read completion interlock will then guarantee that all posted write data has cleared the arbiter.

Another potential issue is relaxed write ordering within a system agent en route to the EDMA destination device. If the hardware were to issue EDMA outbound posted writes and IA-32 core posted writes with differing stream ID codes, an intermediary component may allow the IA-32 core doorbell access to move around posted EDMA data. This would again result in stale data “executed” by the destination device.

The IMCH could solve this problem by issuing an explicit FENCE between the final EDMA write and the interrupt, preventing subsequent IA-32 core accesses from reordering en route to the destination. A simpler (but more limited) solution is to utilize the same stream ID for all outbound traffic regardless of source. This makes transactions initiated by each EDMA channel indistinguishable from those initiated by any of the IA-32 core threads. With no stream ID information to determine reordering legality, an intermediary device must necessarily enforce strong ordering for all accesses outbound.

The single initiator ID in concert with the internal interlock for APIC messages and read completions is sufficient to guarantee proper behavior. The implementation guarantees that any flag write or data read to the destination port will necessarily push the EDMA transfer data ahead of it, ensuring correct producer/consumer operation.

12.11 Initiating an EDMA Transfer

The following subsections detail the steps the software must take in programming a channel to initiate a transfer (or chain of transfers). The steps covered include channel initialization, transfer start, and suspend or stop. Each channel is designed to have independent control of interrupt enabling and generation, and independent transfer attribute controls; this provides the greatest flexibility to the application program.

12.11.1 Setup and Initiation

Initializing a channel begins with constructing one or more chain descriptors in local system memory. Each chain descriptor takes the form described in Section 12.3.1, “Chain Descriptor Definition” on page 311. Once the descriptors are defined, the following steps are required to initiate a transfer:

1. Ensure the EDMA channel is enabled and in Normal Mode by setting the EDMA Enable and Mode bits of the EDMA Control Register.
2. The channel must be inactive/idle prior to starting a transfer. This may be verified by reading the Channel Active bit in the Channel Status Register (CSR), which is clear when the channel is inactive/idle.
3. Update the Next Descriptor Address Register (NDAR/NDUAR) with the address of the first chain descriptor in local system memory.

4. Clear the Channel Status Register (CSR) of any asserted error or status bits. Each EDMA channel will not initiate a new transfer when an error condition remains in the CSR.
5. Clear the Suspend bit and set the Start bit in the Channel Control Register (CCR). Since this is the start of a new transfer and not the resumption of a previous transfer, the Channel Resume bit in the CCR must be clear. (Resume overrides start.)
6. The channel starts the transfer by fetching the chain descriptor at the address contained in the Next Descriptor Address Register (NDAR/NDUAR). The channel moves the NDAR/NDUAR values into the CDAR/CDUAR, and loads the chain descriptor values into their corresponding internal registers. If the load completes without any error, the actual data transfer begins. The Current Descriptor Address Register (CDAR/CDUAR) now contains the address of the chain descriptor just fetched and the Next Descriptor Address Register (NDAR/NDUAR) now contains the descriptor address of the next descriptor in the chain, if any.
7. When the current EDMA transfer has completed without any errors, the channel fetches the next chain descriptor from the address contained in the Next Descriptor Address Register (NDAR/NDUAR) automatically without any software intervention, and proceeds with the next block transfer (provided the value in the NDAR/NDUAR pair is non-zero).

The last descriptor in the chain list has a null value in the Next Descriptor Address field, specifying the end of the chain. The null value in the Next Descriptor Address Register (NDAR/NDUAR) notifies the channel not to read additional chain descriptors from local system memory, and the channel goes idle.

12.11.2 Suspend Function

Software may temporarily suspend execution of a descriptor chain by setting the Suspend bit in the Channel Control Register (CCR). The target channel will complete execution of the current descriptor and suspend operation without losing current status. Software may later cause the channel to resume execution of the descriptor chain by writing to the CCR to clear the Suspend bit and set the Resume bit. In response, the channel will initiate a descriptor fetch from the NDAR/NDUAR, and resume the suspended operation.

Software does not need to re-program the channel configuration after a suspend sequence.

12.11.3 Stop Function

Software may intentionally abort a transfer by setting the Stop bit in the Channel Control Register (CSR). Once aborted, the transfer cannot be resumed. In response to the Stop bit, the target channel will immediately cease fetching source data, drain any buffered destination data, and go idle. Usage of this mechanism will result in assertion of the Stopped status bit in the CSR, and will generate an interrupt if so enabled. (Note that the stop function is sufficiently fast in the IMCH that reading back the channel status register is preferred over utilizing the interrupt on stop function.)

Usage of the Stop mechanism will not result in assertion of the Abort status bit, nor will it generate an “interrupt on abort” indication if so enabled. The channel differentiates an abort on error from an abort on software command, and will ensure that all error status bits remain clear.

If the MSI mechanism is in use for interrupt generation, and independent messages are defined for abort on error and normal run-time interrupts, the latter message type will be utilized on behalf of the stop function.

12.11.4 EDMA Process Flow

Figure 12-17 provides a high-level flow chart of the EDMA initialization sequence. Figure 12-18 provides a similar view of the EDMA completion sequence.

Figure 12-17. Initiation Flow Chart

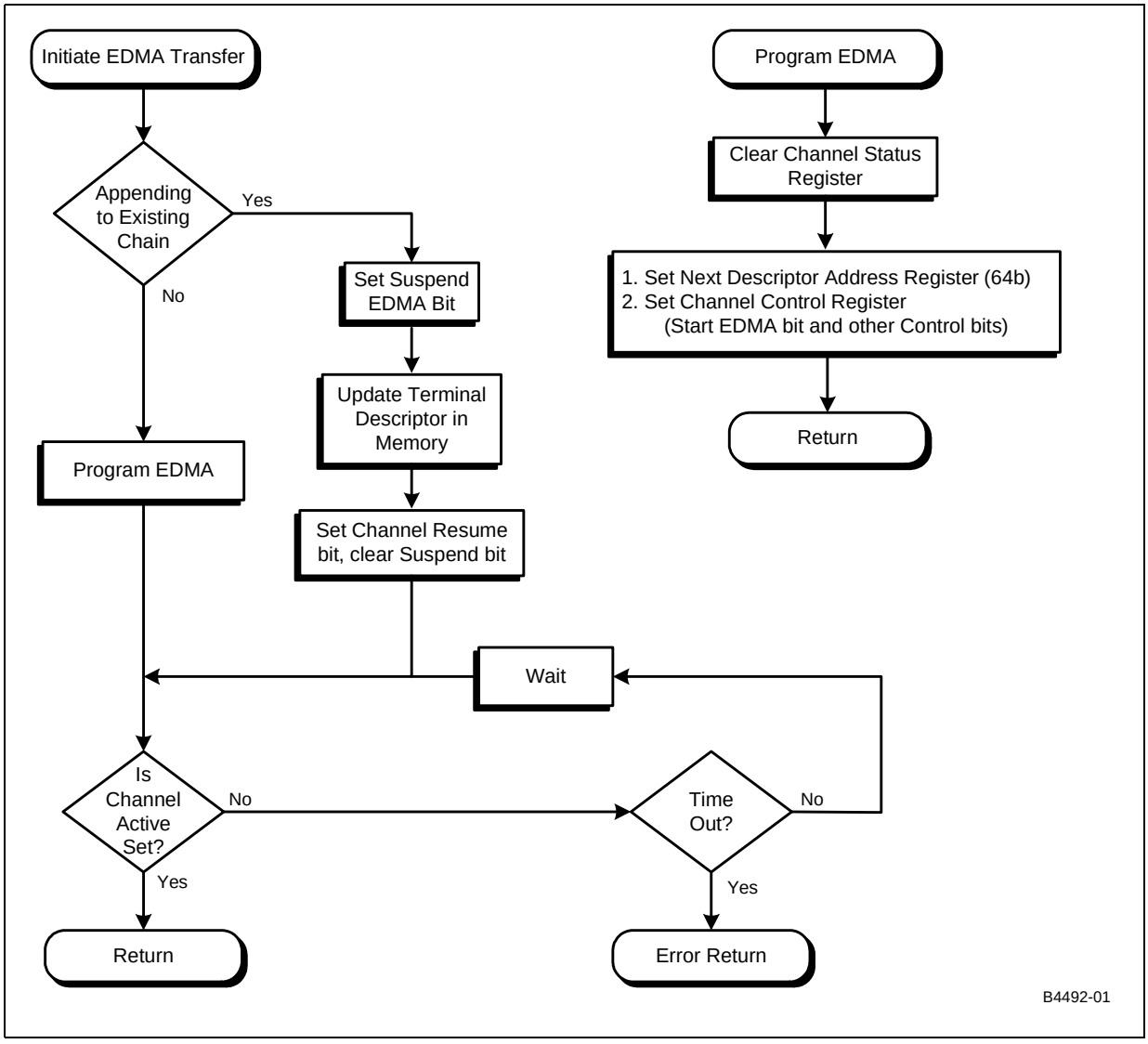
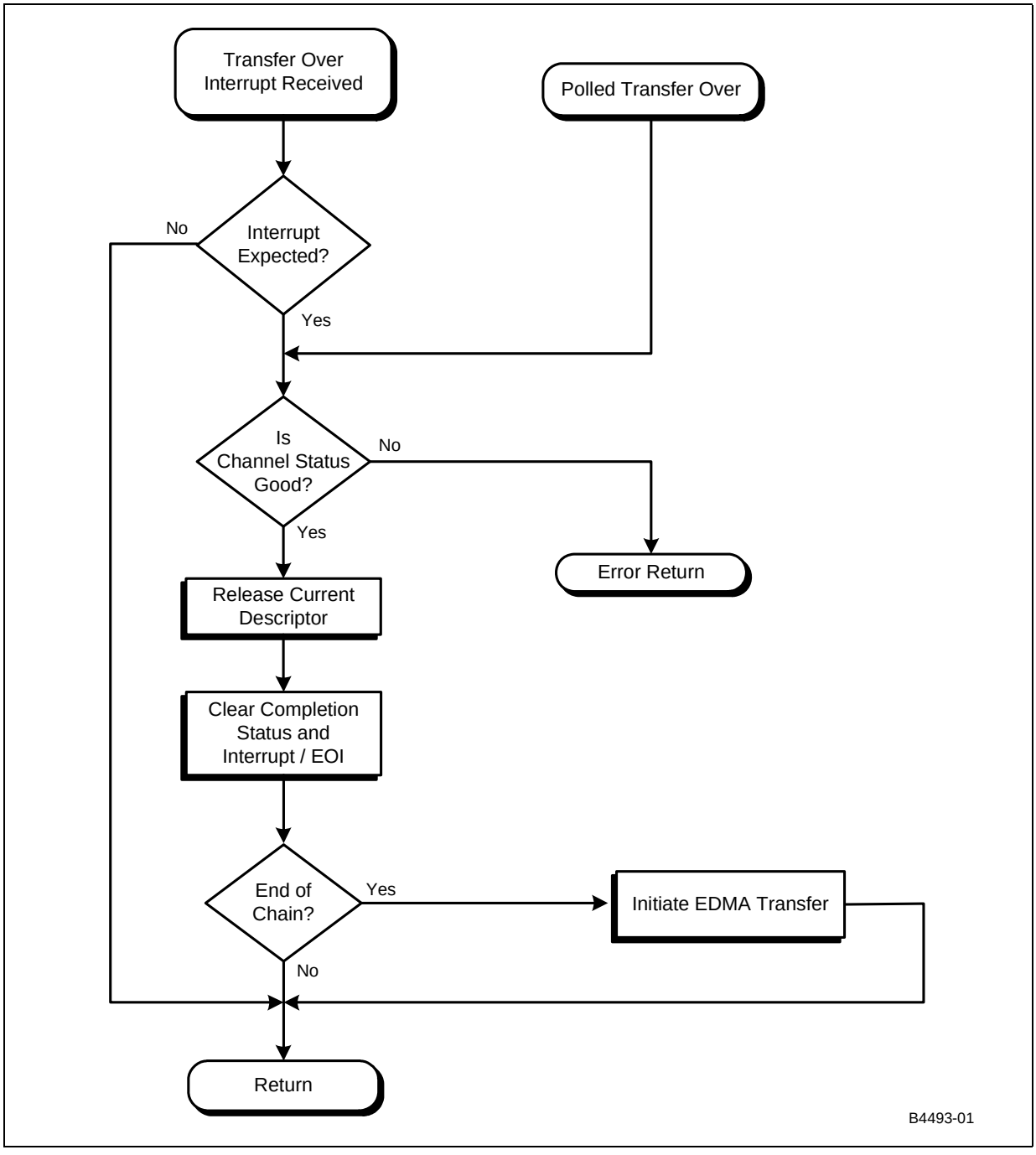


Figure 12-18.Completion Flow Chart



§ §

13.0 Platform Configuration

13.1 RASUM Features - SMBus Access

Configuration registers are accessible from either the IA-32 core or from the SMBus. The IA-32 core will be able to access all configuration registers through host configuration cycles. Access via SMBus is read/write to the IMCH configuration registers. The SMBus cannot use the IMCH's SM-port target interface to access any register in the IICH or outside of CMI. Each device must have its own SMBus target port.

CMI does not shadow the RASUM registers for the SMBus. To clear these registers, a write access will need to be performed. The IMCH SMBus has full read/write access to the IMCH PCI legacy registers.

The IMCH global RASUM register set and those registers applicable to logical bus#0 and memory are implemented in Function 1 of Device 0. RASUM registers specific to other internal devices appear in the register map for the associated device. The IMCH error control registers are in Function 1, and are read/write accessible by the processor and through the SMBus. The IMCH error logging registers are also available to the processor and SMB master in Function 1. The IMCH RASUM control register and the “CMD” registers (SERRCMD, SMICMD, etc.) which control generation of SERR#, SMI#, and SCI# are read/write accessible by the processor and through the SMBus.

FSB-initiated accesses to configuration space registers are serviced through configuration ring. It is perfectly legal for an SMBus access to be requested while an FSB-initiated access is already in progress. In other words, SMBus configuration accesses and processor configuration cycles may occur at the same time. The IMCH supports “wait your turn” arbitration to resolve all collisions and overlaps, such that the access that reaches the configuration ring arbiter first is serviced first while the conflicting access is held off. An absolute tie at the arbiter is resolved in favor of the FSB.

13.2 Platform Configuration Structure Conceptual Overview

The IMCH and IICH are physically connected by an internal interface called NSI (North South Interface). From a configuration standpoint, NSI is logically PCI bus #0. As a result, all devices internal to the IMCH and IICH, except host switch devices appear to be on PCI bus #0. The system's primary PCI expansion bus is physically attached to the IICH and, from a configuration perspective, appears to be a hierarchical PCI bus behind a PCI-to-PCI bridge and therefore has a programmable Bus number. The PCI Express ports appear to system software to be real PCI buses behind PCI-to-PCI bridges that reside as devices on PCI bus #0.

CMI decodes multiple PCI Device numbers. The configuration registers for the devices are mapped as devices residing on PCI bus #0 except for host switch devices. Each Device Number may contain multiple functions. See Table 13-1, “PCI Devices and Functions on Bus 0” for device and function assignments.

Table 13-1. PCI Devices and Functions on Bus 0

Device	Function	Function Description
0	0	IMCH
0	1	IMCH, error status
1	0	IMCH EDMA engine
2	0	IMCH PCI Express Port A0 X8 or X4 unit
8	0	IMCH Test and Device 0 Overflow
31	0	IICH LPC Interface
31	2	IICH SATA Controller
31	3	IICH SMBus Controller
31	5	Reserved
31	6	Reserved
29	0	IICH USB Controller #1
29	7	IICH USB 2.0 Controller

13.2.1 IMCH PCI Devices

The PCI predefined header has five fields that deal with device identification. All devices are required to implement these fields. Generic configuration software is able to easily determine the device available for use. These registers are read only. The five fields are vendor ID, device ID, revision ID, header type, and class code:

- The 16-bit vendor ID is assigned by PCI SIG and has a value of 8086h for Intel.
- The 16-bit device ID is assigned by the vendor.
- The 8-bit revision ID is chosen by the vendor to indicate the different steppings of a device. The value 00h designates an A0 stepping. The value 01h designates an B0 stepping.
- The header type specifies the structure of the second half of the header, and also whether or not the device has multiple functions. The value 80h indicates a multi-function device.
- The class-code field identifies the generic function of the device. The class-code is further broken into three sub-fields, base class, sub-class, and programming interface. CMI proper has a base class code of 06h indicating a bridge device. The sub-class value of 00h indicates a host bridge.

A disabled or non-existent IMCH device's configuration register space is hidden, returning all 1's for reads and dropping writes just as if the cycle terminated with a Master Abort on PCI.

If one or more IICH devices or some of their functions are not supported on the platform, each can be disabled individually. When a device or function is disabled, it does not appear at all to the software: No responses to any register reads and no responses to any register writes. This is intended to prevent software from thinking that a device or function is present (and reporting it to the end-user).

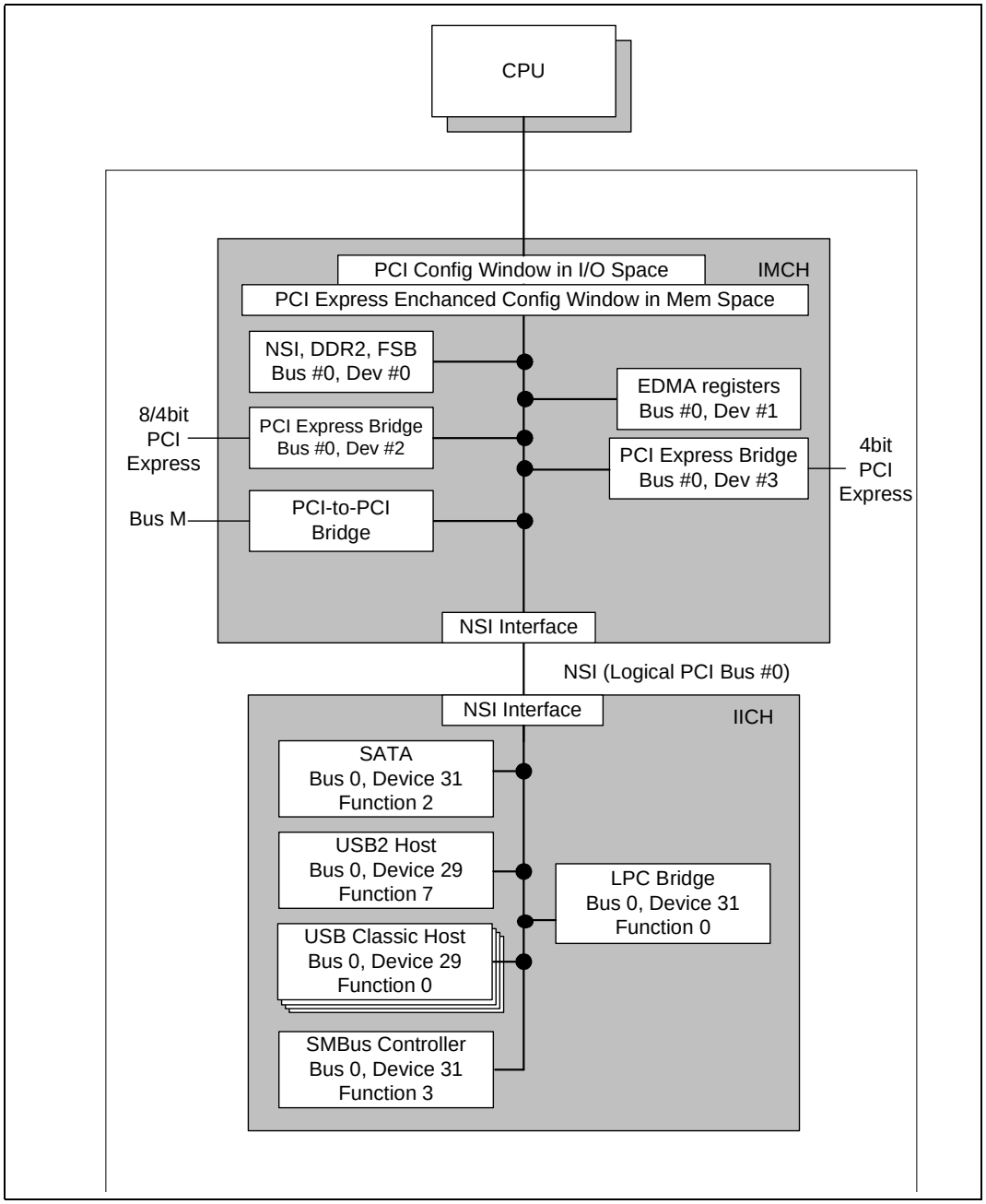
When a PCI Express interface is unpopulated or fails to train, the associated configuration register space is hidden, returning all ones for all registers just as if the cycle terminated with a Master Abort on PCI. Also, if PCI Express port PEA0 is configured for x8 operation rather than x4, the corresponding PCI Express port PEA1 configuration space will be hidden.

13.2.2 IICH PCI Devices

Logically, the IICH appears as multiple PCI devices within a single physical component also residing on PCI bus #0. One of the IICH devices is a PCI-to-PCI bridge. Logically, the primary side of the bridge resides on PCI #0 while the secondary is a standard PCI expansion bus.

Note: The internal devices in the IMCH and IICH (except host switch devices) logically constitute as PCI Bus #0 to configuration software (see Figure 13-1).

Figure 13-1. Bus 0 Device Map



13.3 Routing Configuration Accesses

The IMCH supports up to two x4 PCI Express interfaces:

- PEA0
- PEA1
  These two interfaces can be combined to form a x8 interface, PEA.

The IMCH is responsible for routing PCI configuration cycles to the proper interface. PCI configuration cycles to IICH internal devices and downstream devices are routed to the IICH via the internal NSI bus. PCI configuration cycles to the IMCH PCI Express interfaces are routed to PEA(0:1). Routing of configuration accesses to PEA(0:1) is controlled via the standard PCI-to-PCI bridge mechanism using information contained within the PRIMARY BUS NUMBER, the SECONDARY BUS NUMBER, and the SUBORDINATE BUS NUMBER registers of the corresponding PCI-to-PCI bridge device.

A detailed description of the mechanism for translating IA-32 core I/O bus cycles to configuration cycles on one of the buses is described below.

Note: The IMCH supports a variety of connectivity options. When any of the IMCH's interfaces are disabled, the associated interface's device registers are hidden. All configuration cycles (reads and writes) to disabled devices on bus #0 are forwarded to the NSI where they will Master Abort.

13.3.1 Standard PCI Bus Configuration Mechanism

The PCI Bus defines a slot based "configuration space" that allows each device to contain up to eight functions with each function containing up to 256 8-bit configuration registers. The *PCI Specification* defines two bus cycles to access the PCI configuration space: Configuration Read and Configuration Write. Memory and I/O spaces are supported directly by the IA-32 core. Configuration space is supported by a mapping mechanism implemented within the IMCH. The *PCI Specification* defines two mechanisms to access configuration space, Mechanism #1 and Mechanism #2. CMI supports Mechanism #1.

The configuration access mechanism makes use of the CONFIG_ADDRESS Register and CONFIG_DATA Register. To reference a configuration register a Dword (32-bit) I/O write cycle is used to place a value into CONFIG_ADDRESS that specifies the PCI bus, the device on that bus, the function within the device, and a specific configuration register of the device function being accessed. CONFIG_ADDRESS[31] must be a '1' to enable a configuration cycle. CONFIG_DATA then becomes a window into the four bytes of configuration space specified by the contents of CONFIG_ADDRESS. Any read or write to CONFIG_DATA will result in the IMCH translating the CONFIG_ADDRESS into the appropriate configuration cycle.

The IMCH is responsible for translating and routing the IA-32 core I/O accesses to the CONFIG_ADDRESS and CONFIG_DATA registers to internal IMCH configuration registers, for NSI, and PCI Express ports PEA(0:1).

Note: It is only possible to generate 1-4 byte configuration accesses via this mechanism, which is in line with IMCH capabilities. The IMCH ***ONLY*** supports accesses up to 1 Dword (32 bits) in size into the configuration register space (internal or external).

13.3.2 PCI Bus #0 Configuration Mechanism

The IMCH decodes the Bus Number (bits 23:16) and the Device Number fields of the CONFIG_ADDRESS register. If the Bus Number field of CONFIG_ADDRESS is 0, the configuration cycle is targeting a PCI Bus #0 device.

The Host-NSI Bridge entity within the IMCH is hardwired as Device #0 on PCI Bus #0.

The EDMA Controller within the IMCH is hardwired as Device #1 on PCI Bus #0.

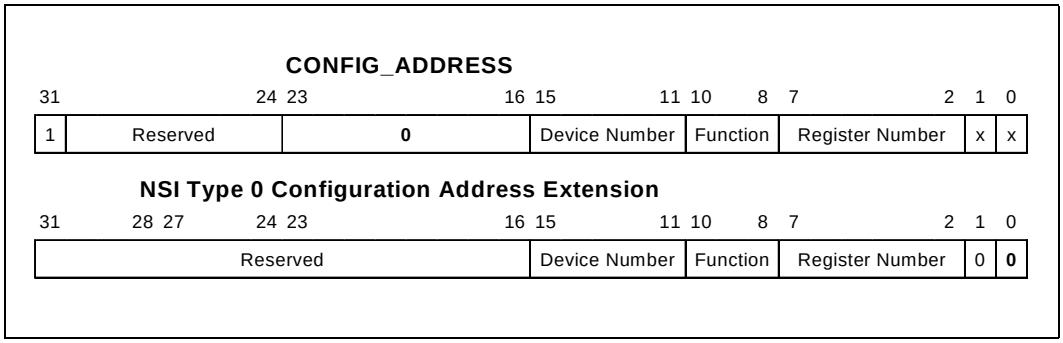
The Host-PEA0 bridge entity within the IMCH is hardwired as Device #2 on PCI Bus #0.

The Host-PEA1 bridge entity within the IMCH is hardwired as Device #3 on PCI Bus #0.

The PCI-to-PCI Bridge entity within the IMCH is hardwired as Device #4 on PCI Bus #0

Configuration cycles to any of the IMCH's enabled internal devices are confined to the IMCH and not sent over NSI. Accesses to disabled IMCH internal devices, or devices #10 to #31 is forwarded over NSI as Type 0 Configuration Cycles. A[1:0] of the NSI Request Packet for the Type 0 configuration cycle is "00". Bits 31:2 of the CONFIG_ADDRESS register is translated to the A[31:2] field of the NSI Request Packet of the configuration cycle as shown in Figure 13-2. The IICH decodes the Type 0 access and generates a configuration access to the selected internal device.

Figure 13-2. NSI Type 0 Configuration Address Translation

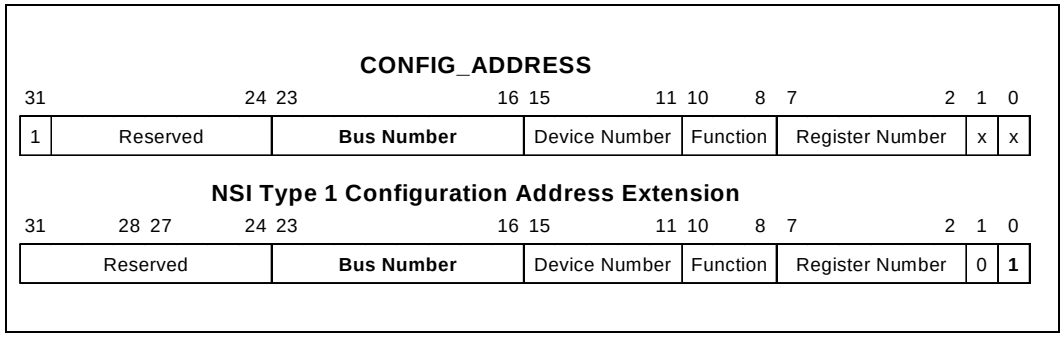


13.3.3 Primary PCI and Downstream Configuration Mechanism

If the Bus Number in the CONFIG_ADDRESS is non-zero, and does not lie between the SECONDARY BUS NUMBER register and the SUBORDINATE BUS NUMBER register for one of the PCI Express ports, the IMCH will generate a Type 1 NSI Configuration Cycle. A[1:0] of the NSI request packet for the Type 1 configuration cycle is "01". Bits 31:2 of the CONFIG_ADDRESS register is translated to the A[31:2] field of the NSI request packet of the configuration cycle as shown in Figure 13-3. This NSI configuration cycle is sent over NSI.

If the cycle is forwarded to the IICH via NSI, the IICH compares the non-zero Bus Number with the SECONDARY BUS NUMBER and SUBORDINATE BUS NUMBER registers of its P2P bridges to determine if the configuration cycle is meant for the Primary PCI or one of the IICH's PCI Express ports.

Figure 13-3. NSI Type 1 Configuration Address Translation

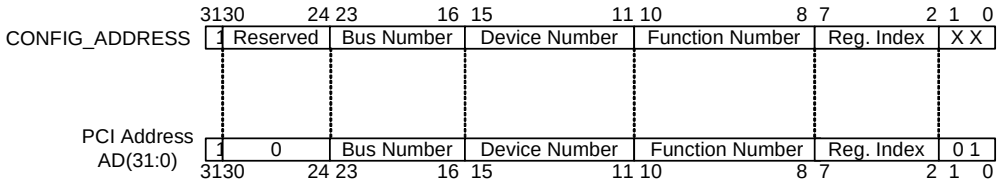


13.3.4 IMCH PCI Express Bus Configuration Mechanism

From the configuration perspective, the PCI Express ports are seen as PCI bus interfaces residing on a Secondary Bus side of the "virtual" PCI-to-PCI bridges referred to as the IMCH Host-PCI Express bridge. On the Primary bus side, the "virtual" PCI-to-PCI bridge is attached to PCI Bus #0. Therefore the PRIMARY BUS NUMBER register is hardwired to "0". The "virtual" PCI-PCI bridge entity converts Type #1 PCI Bus Configuration cycles on PCI Bus #0 into Type 0 or Type 1 configuration cycles on the PCI Express interfaces. Type 1 configuration cycles on PCI Bus #0 that have a BUS NUMBER that matches the SECONDARY BUS NUMBER of one of the IMCH's "virtual" P2P bridges are translated into Type 0 configuration cycles on the appropriate PCI Express interface. The address bits are mapped as described in Figure 13-3.

If the Bus Number is non-zero, greater than the value programmed into the SECONDARY BUS NUMBER register, and less than or equal to the value programmed into the corresponding SUBORDINATE BUS NUMBER register the configuration cycle is targeting a PCI bus downstream of the targeted PCI Express interface. The IMCH will generate a Type 1 configuration cycle on the appropriate PCI Express interface. The address bits are mapped as described in Figure 13-4.

Figure 13-4. Mechanism #1 Type 1 Configuration Address to PCI Address Mapping



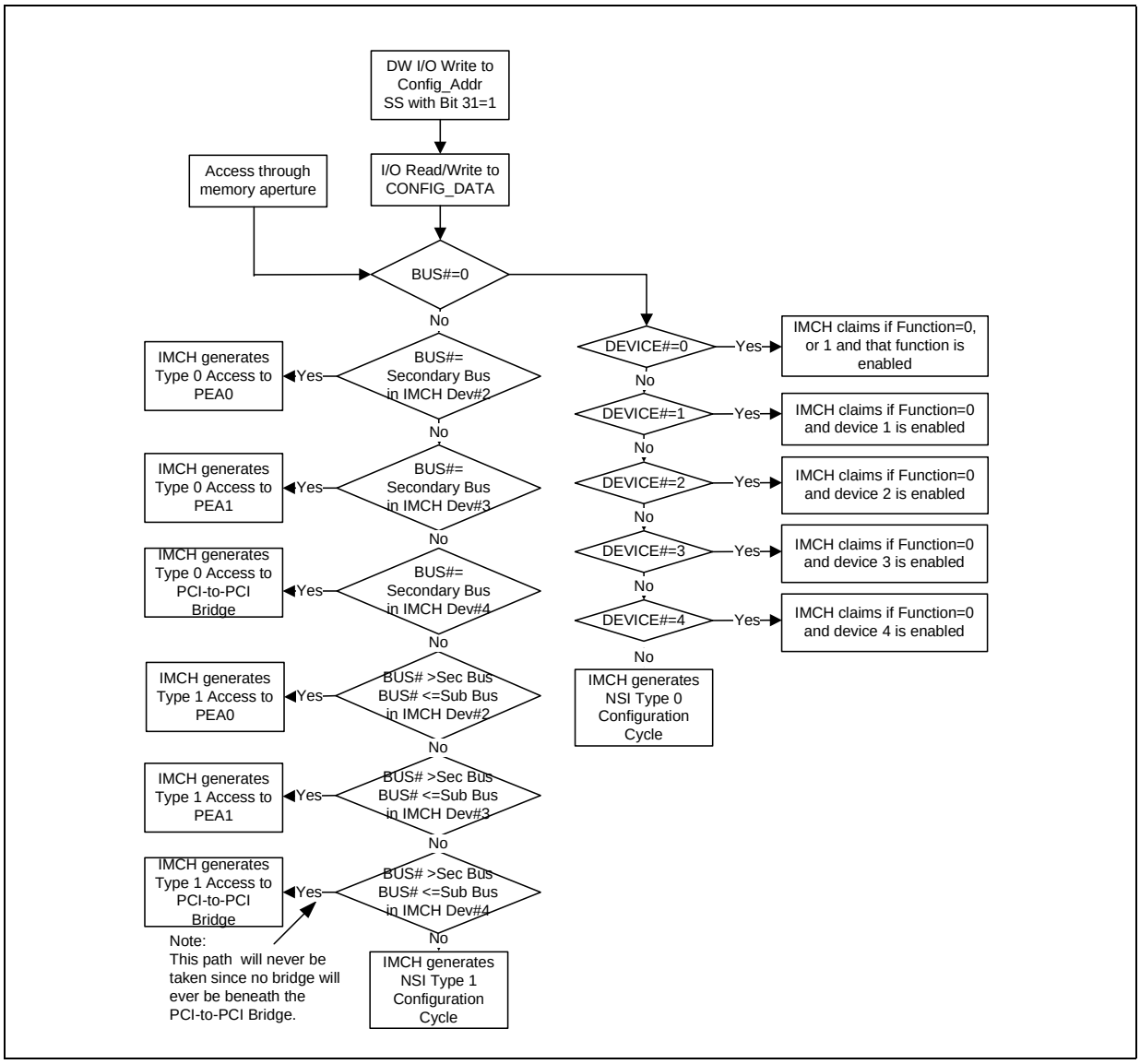
To prepare for mapping of the configuration cycles on PCI Express the initialization software will go through the following sequence:

Scan all devices residing on the PCI Bus #0 using Type 0 configuration accesses.

For every device residing at bus #0 which implements PCI-to-PCI bridge functionality, it will configure the secondary bus of the bridge with the appropriate number and scan further down the hierarchy. This process will include the configuration of the "virtual" PCI-to-PCI bridges within the IMCH used to map the PCI Express device's address spaces in a software specific manner.

13.3.5 IMCH Configuration Cycle Flow Chart

Figure 13-5. IMCH Configuration Flow Chart



13.4 IMCH Register Introduction

The IMCH contains two sets of software accessible registers, accessed via the IA-32 core I/O address space: control registers I/O mapped into the IA-32 core I/O space, which control access to PCI configuration space, and internal configuration registers residing within the IMCH, which are partitioned into multiple logical device register sets (“logical” since they reside within a single physical device).

The IMCH internal registers (I/O Mapped and Configuration registers) are accessible by the IA-32 core. The registers can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities, with the exception of CONFIG_ADDRESS, which can only be accessed as a Dword. All multi-byte numeric fields use “little-endian” ordering (i.e., lower addresses contain the least significant parts of the field).

Note: Irrespective of the access mechanism used (I/O register mechanism, or memory-mapped mechanism), the IMCH ***ONLY*** supports 1-4 byte accesses into configuration space. Software must (if necessary) take steps to prevent use of opcodes that would treat configuration space destinations as objects greater than a single Dword (32 bits) in size. Such attempted usage will result in spurious behavior up to and including hanging the platform.

Some of the IMCH registers described in this section contain reserved bits which are labeled “Reserved”. Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back. Note the software does not need to perform read, merge, write operation for the configuration address register.

In addition to reserved bits within a register, the IMCH contains address locations in the configuration space of the Host-NSI Bridge entity that are marked either “Reserved” or “Intel Reserved”. The IMCH responds to accesses to “Reserved” address locations by completing the host cycle. When a “Reserved” register location is read, a zero value is returned. (“Reserved” registers can be 8-, 16-, or 32-bit in size). Write operations to “Reserved” registers have no effect on the IMCH. Registers that are marked as “Intel Reserved” must not be modified by system software. Writes to “Intel Reserved” registers may cause system failure. Reads to “Intel Reserved” registers may return a non-zero value.

Upon a Reset, CMI sets its entire internal configuration registers to predetermined default states. At reset, some register values are determined by external strapping options. A register’s default value represents the minimum functionality feature set required to successfully bring up the system. It is the responsibility of the system initialization software (usually the BIOS) to properly determine the DRAM configurations, operating parameters and optional system features that are applicable, and to program CMI registers accordingly.

13.5 IMCH Sticky Registers

Certain registers in the IMCH are sticky through a hard-reset. They will only be reset on a Power-good reset. In general, these registers are the error logging registers and a few special cases. The error command registers are not sticky, so that on reset bogus errors are not reported and that errors are not reported through a mechanism that hasn’t been set up in code yet. Only those registers that are explicitly marked as “Sticky: YES” are sticky. Those not marked or those marked as NO are not sticky.

The following registers are sticky:

- Device 0, Function 0: Critical DRAM control registers, a portion of DRC, DRT clock gearing and clock disable registers
- Device 0, Function 0: ECO sticky register
- Device 0, Function 0, Bar 14: BIOS notepad sticky register
- Device 0, Function 1: error information registers (Not the command registers)
- Device 2, Function 0: error information registers (Not the command registers)
- Device 3, Function 0: error information registers (Not the command registers)
- Device 8, Function 0: PILOT control registers
- Device 8 Function 0: Power On Configuration bits

13.6 IMCH I/O Mapped Registers

The IMCH contains two registers that reside in the IA-32 core I/O address space – the Configuration Address (CONFIG_ADDRESS) Register and the Configuration Data (CONFIG_DATA) Register. The Configuration Address Register enables/disables the configuration space and determines what portion of configuration space is visible through the Configuration Data window.

Table 13-2. Summary of IMCH PCI Configuration Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
0CF8h	0CF8h	"Offset 0CF8h: CONFIG_ADDRESS: Configuration Address Register" on page 354	00000000h
0CFCh	0CFCh	"Offset 0CFCh: CONFIG_DATA: Configuration Data Register" on page 355	00000000h

13.6.0.1 Offset 0CF8h: CONFIG_ADDRESS - Configuration Address Register

CONFIG_ADDRESS is a 32-bit register that can be accessed only as a Dword. A Byte or Word reference will "pass through" the Configuration Address Register and NSI onto the IICH as an I/O cycle. The CONFIG_ADDRESS register contains the Bus Number, Device Number, Function Number, and Register Number for which a subsequent configuration access is intended.

Table 13-3. Offset 0CF8h: CONFIG_ADDRESS: Configuration Address Register (Sheet 1 of 2)

Description:						
View: IA F	**Base Address:** 0000h (IO)				**Offset Start:** 0CF8h **Offset End:** 0CF8h	
Size: 32 bit	**Default:** 00000000h				**Power Well:** Core	
Bit Range		**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
	31	CFGE	**Configuration Enable.** 0 = Accesses to PCI configuration space are disabled. 1 = Accesses to PCI configuration space are enabled.		0h	RW
30	:24	Reserved	**Reserved.** These bits are read only and have a value of 0.		0h	RO
23	:16	Bus_Number	Contains the bus number being targeted by the configuration cycle.		0h	RW

Table 13-3. Offset 0CF8h: CONFIG_ADDRESS: Configuration Address Register (Sheet 2 of 2)

Description:							
View: IA F		**Base Address:** 0000h (IO)				**Offset Start:** 0CF8h **Offset End:** 0CF8h	
Size: 32 bit		**Default:** 00000000h				**Power Well:** Core	
Bit Range		**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15	:11	Device_Number	Selects one of the 32 possible devices per bus.			0h	RW
10	:08	Function_Number	Selects one of eight possible functions within a device.			0h	RW
07	:02	Register_Number	This field selects one register within the particular Bus, Device, and Function as specified by the other fields in the Configuration Address Register. This field is mapped to A[07:02] during NSI or PCI Express Configuration cycles.			0h	RW
01	:00	Reserved	**Reserved**			0h	

13.6.0.2 Offset 0CFCh: CONFIG_DATA - Configuration Data Register

CONFIG_DATA is a 32-bit read/write window into configuration space. The portion of configuration space that is referenced by CONFIG_DATA is determined by the contents of CONFIG_ADDRESS.

Table 13-4. Offset 0CFCh: CONFIG_DATA: Configuration Data Register

Description:							
View: IA F		**Base Address:** 0000h (IO)				**Offset Start:** 0CFCh **Offset End:** 0CFCh	
Size: 32 bit		**Default:** 00000000h				**Power Well:** Core	
Bit Range		**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31	:00	CDW	**Configuration Data Window.** If bit 31 of CONFIG_ADDRESS is one any I/O access to the CONFIG_DATA register is mapped to configuration space using the contents of CONFIG_ADDRESS.			0h	RW

13.7 IMCH Memory Mapped Registers

Certain DRAM compensation control, EDMA control/status registers, NSI control/status and PCI Express will reside in memory mapped space instead of configuration space. These memory mapped address regions are setup through base address registers and capability pointers, which will reside in configuration address space. These registers are documented in the configuration register chapter. These base address registers follow the standard definition as found in the *PCI Express Specification*.

These memory mapped register regions must not be marked as WC (Write-Combining), as all accesses to the registers within these regions are limited to Dword access, and write-combining is not allowed. Further, these registers must not be accessed utilizing IA-32 core operations with a data operand size greater than 32-bits, as such access is strictly unsupported by the IMCH.

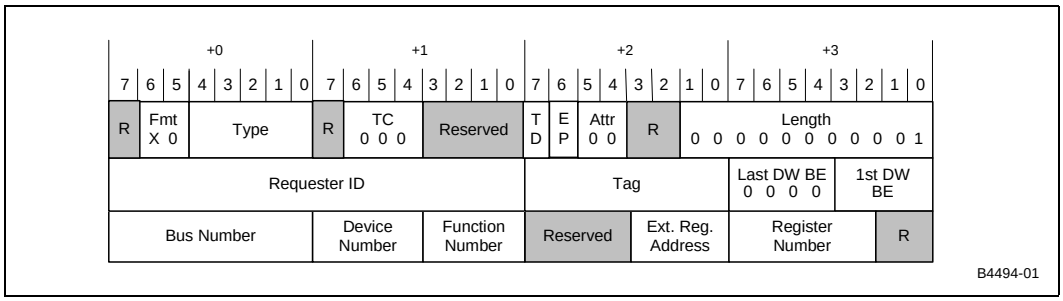
13.8 PCI Express Enhanced Configuration Mechanisms

PCI Express extends the configuration space to 4096 bytes per device/function as compared to 256 bytes allowed by PCI 2.2 configuration space. PCI Express configuration space is divided into a PCI 2.2 compatible region, which consists of the first 256 B of a logical device's configuration space and an extended PCI Express region which consists of the remaining configuration space. The PCI 2.2 compatible region can be accessed using either the mechanisms defined in the PCI 2.2 or using the enhanced PCI Express configuration access mechanism. All changes made using either access mechanism are equivalent; however, software is not allowed to interleave PCI Express and PCI access mechanisms to access the configuration registers of devices. The extended PCI Express region can only be accessed using the enhanced PCI Express configuration access mechanism.

13.8.1 PCI Express Configuration Transaction Header

The PCI Express Configuration Transaction Header includes an additional four bits for the Register Number field (ExtendedRegisterAddress[3:0]) to provide additional configuration space.

Figure 13-6. PCI Express Configuration Transaction Header



The PCI 2.2 compatible configuration access mechanism uses the same Request format as the enhanced PCI Express mechanism. For PCI compatible Configuration Requests, the Extended Register Address field must be all zeros.

To maintain compatibility with PCI configuration addressing mechanisms, system software must access the enhanced configuration space using Dword operations (Dword-aligned) only.

13.8.2 Enhanced Configuration Hardware Implications

The IMCH must translate the memory-mapped extended enhanced PCI Express configuration access cycles from the host processor to PCI Express configuration cycles.

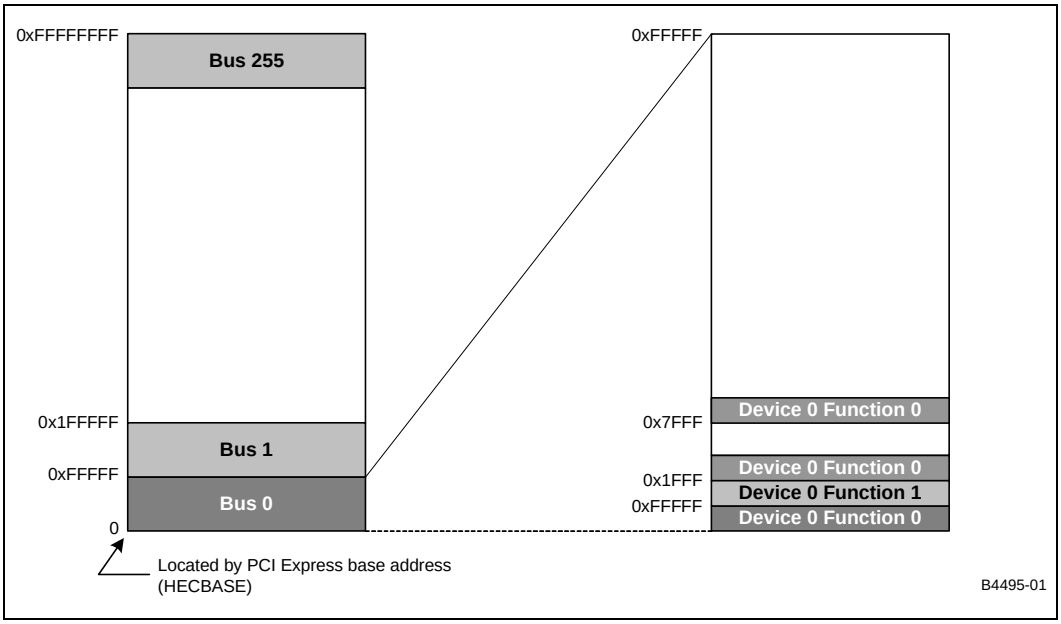
Devices are required to respond to an additional four bits for decoding configuration register access. Devices must decode the ExtendedRegisterAddress[3:0] field of the Configuration Request Header. This field is used in conjunction with the Register Number to specify the Dword address of the register being accessed.

A PCI Express device must be able to operate with basic required functionality in a legacy environment without requiring access to any extended PCI Express configuration.

13.8.3 Enhanced Configuration Memory Address Map

The Enhanced Configuration Memory Address Map is positioned into CMI memory space by use of the PCI Express Enhanced Configuration Base register known as HECBASE. This register contains the address that corresponds to bits 31 to 28 of the base address for PCI Express enhanced configuration space below 4 GB. Configuration software will read this register to determine where the 256 Mbyte range of memory addresses resides for enhanced configuration. This register defaults to a value of E, which corresponds to E000 0000 for the IMCH. It is not intended that this value is ever changed by BIOS.

Figure 13-7. Enhanced Configuration Memory Address Map



13.8.4 Enhanced Configuration FSB Address Format

Table 13-5 presents the enhanced configuration address format for the front side bus. Note that bits 31:28 of Table 13-5 correspond to the default value of HECBASE.

Table 13-5. Enhanced Configuration FSB Address Format

Bits	Description
35:32	0h
31:28	Eh
27:20	Bus Number
19:15	Device Number
14:12	Function Number
11:00	Register Offset

§ §

14.0 RAS Features and Exception Handling

CMI is designed to bring enterprise level reliability, availability, serviceability, usability, and manageability (RASUM) to the embedded platform.

14.1 RAS Features

14.1.1 Data Protection

Due to the nature of having various data protection schemes on the different interfaces (ECC, parity, and CRC) it is necessary to be able to convert between them when transferring data internally. To accomplish this, protection of internal data is done with parity.

14.1.1.1 DRAM ECC

The DRAM interface uses a standard SEC/DED ECC across a 64-bit data quantity.

14.1.1.2 PCI Express Interface

These high-speed serial interfaces have traditional CRC protection. The data packets utilize a 32-bit CRC protection scheme, specifically the same CRC-32 used by Ethernet - 0x04C11DB7. The smaller and less error-prone link packets utilize a 16-bit CRC scheme. Since packets utilize 8B/10B encoding and not all encodings are used, this provides further data protection because illegal codes can be detected. Also, if errors are detected on the reception of data packets due to various transients, these data packets can be retransmitted. Hardware logic supports this link-level retry without software intervention.

14.1.1.3 Data Error Propagation Between Interfaces/Units

Due to the nature of having various data protection schemes; ECC, parity, and CRC - it is necessary to be able to convert between the separate schemes. Beyond this requirement, it is necessary to indicate whether or not incoming data is corrupted. Also, it is useful to know when internal data has been corrupted during transit. To accomplish this, the IMCH uses parity to protect internal data. This requires units to add two parity bits for each 64 bits of data path width. Data received by a unit from outside the chip creates two parity bits to travel with the data, one provides parity on the upper 32 bits, and the other provides parity on the lower 32 bits. If either of the 32-bit halves is required to be poisoned, both halves are poisoned. This provides the user of the data a mechanism to recognize when a bit was flipped in transit by detecting when only one of the parity bits is bad. The user will flag this error condition as well as mark both halves bad. This covers both cases of the data starting out as either good or bad. If it started out as good, but a bit was flipped, it is indeed corrupted and must be marked as such. In the case where it started out as bad, and a bit was flipped, it is still corrupted, although probably a different data value than its starting value. This scheme works when all quantities being passed are 64 bits or greater. If a data path must be padded, it must be padded with zeroes. Even parity will be used for

this scheme; meaning that the total number of asserted bits including the parity bits is an even number of bits. This parity protection scheme applies to different interfaces on the chip hence the name: “Chip Two Bit Parity” or CTB parity.

Note: Due to EDMA byte realignment and parity manipulation, a single CTB parity bit error observed by the EDMA unit may poison either 2 or 4 DWords depending on the resultant alignment. Refer to the EDMA chapter for more details.

14.1.2 DRAM Data Integrity

14.1.2.1 Periodic Memory Scrubbing

When enabled a special DRAM memory scrubbing unit will walk through all DRAM, on a periodic basis, doing reads. Correctable errors found by the read are corrected and then the good data written back to DRAM. A write is only performed when a single bit error has been detected and is correctable, except when an incoming write to the same memory address is detected. In this case the scrub write is dropped and the scrub counter is advanced since this location is already being written. These transactions are treated as non-coherent, since these addresses are not placed on the FSB.

The scrub unit starts at an address than can be programmed and counts to 0. The scrub rate is also programmable so using this method, a 4 GB system can be completely scrubbed in less than a day. The cumulative effect of these scrub writes do not cause any noticeable degradation to memory bandwidth, although they will cause a greater latency for that one very infrequent read that is delayed due to the scrub write cycle.

14.1.2.2 DRAM Hardware Initialization

Hardware will be used to initialize main memory under the direction of BIOS. Once BIOS has programmed the IMCH with the DIMM profile, and has configured and calibrated the IMCH and populated DIMMs, it can utilize the MBIST CSRs to initialize and/or test populated memory. The initialization of MBISTCSR will traverse the target range of memory addresses as rapidly as possible, providing an order of magnitude performance improvement over IA-32 core-generated initialization or test.

The MBIST engine can be configured to choose values other than zero. The eight fixed hex data values selectable are alternating pairs of 0/F, A/5, 3/C, or 6/9. Alternate modes are provided in which LFSR random data may be used, or software explicitly specifies the full pattern of bits to be written in a collection of MBIST DATA registers with or without a shift after every write. In all cases of pattern based initialization and test, the MBIST function does NOT calculate ECC on the fixed pattern or programmed value to be written across the target address range. Rather, the fixed pattern is extended to cover the data devices as well as the ECC devices in the target DIMM, and a strict bit-wise comparison is utilized to determine whether read-back verification passes or fails.

Once all desired testing has been completed, WHQL requirements dictate that memory be completely initialized to “0” prior to transferring control to the operating system. To accomplish this, BIOS must clear all the MBISTDATA registers and utilize the explicit pattern mode of MBIST. It is possible to initialize memory a rank at a time, or en-masse, at the discretion of BIOS.

14.1.2.3 Uncorrectable Retries

The memory controller will not support uncorrectable retries.

14.1.2.4 DRAM Refresh

As with any DRAM device, the storage element is inherently leaky, and must be recharged periodically to avoid loss of data integrity. Circuitry in the memory subsystem will ensure that refresh cycles occur in a periodic fashion across all active DIMMS to meet the specific DRAM requirements.

14.1.2.5 DDR I/O Hardware Assisted Calibration

To determine read capture timing, hardware assisted calibration logic writes a pattern into memory and then reads the data back with different hardware settings until the optimum timing is found. Such calibration is described in the initialization walk-through provided in the clocking and reset chapter of this document. Hardware provides the capability to tune receive-enable timing, DQS centering within the received data eye (both vertical and horizontal), output drive strength, and receive termination.

14.1.3 PCI Express Data Integrity

The PCI Express interfaces will incorporate several features to make this interface as robust as possible without software intervention.

14.1.3.1 PCI Express Training

To establish a connection between PCI Express endpoints, they both participate in a sequence of steps known as training. This sequence will establish the operational width of the link as well as adjust skews of the various lanes within a link so that the data sample points can correctly take a data sample off of the link. The x4 link pairs capable of collapsing to x8 will first attempt to train independently, and will collapse to a single link at the x8 width upon detection of a single device returning link ID information upstream. Once the number of links has been established, they will negotiate to train at the highest common width, and will step down in its supported link widths in order to succeed in training. The ultimate result may be that the link has trained as a X1 link. Although the bandwidth of this link size is substantially lower than a X8 link or even a X4 link, it will allow communication between the two devices. Software will then be able to interrogate the device at the other end of the link to determine why it failed to train at a higher width, something that would not be possible without support for the X1 link width. It should be noted that width negotiation is only done during training or retraining, but not recovery.

14.1.3.2 PCI Express Retry

The PCI Express interface incorporates a link level retry mechanism. The hardware detects when a transmission packet is corrupted and a retry of that particular packet and all following packets will be performed. Although this will cause a temporary interruption in the delivery of packets, it does so in order to maintain the link integrity.

14.1.3.3 PCI Express Recovery

When numerous errors occur, the hardware may determine that the quality of the connection is in question, and the end points can enter a quick training sequence known as recovery. The width of the connection will not be renegotiated, but the adjustment of skew between lanes of the link may occur. This occurs without any software intervention, but the software may be notified.

14.1.3.4 PCI Express Retrain

If the hardware is unable to perform a successful recovery then the link will automatically revert to the polling state, and initiate a full retraining sequence. This is a drastic event with an implicit reset to the downstream device and all subordinate

devices, and is logged by the IMCH as a “Link Down” error. If escalation of this event is enabled, software is notified of the link DL_DOWN condition. Once software has to be involved, then data will likely be lost, and processes need to be restarted, but this is still preferred to having to shut the system down, or go offline for an extended period of time.

14.1.4 Test/Support Major Buses

14.1.4.1 IICH XOR

The IICH supports XOR chain test mode. This non-functional test mode is a dedicated test mode when the chip is not operating in its normal manner.

14.1.4.2 SMB (IMCH)

Full access to internal configuration registers via the System Management Bus is supported. This will allow a server management card to control system configuration and to read various error/status information. Accesses to devices marked as not present will still be possible through SMB.

14.1.4.3 SMB (IICH)

The IICH SMB is SMBus 2.0 compliant and it is compatible with most 2-wire components that are also I2C compatible. The host interface allows IA-32 core to communicate via SMBus, the slave interface allows external microcontroller to access system resource in IICH. This IICH SMB does not support access to internal configuration registers.

14.1.4.4 I^2C

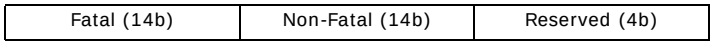
Access to the external DIMMs will be through the IICH, via I^2C. This will be used to determine the nature of the DIMMs present in order to configure the memory subsystem correctly.

14.2 Exception Handling

There are a variety of exception conditions. Some are internally detected; some are detected on input pins; some are passed on behalf of other devices. All recognized exceptions eventually cause the IMCH to do one of the following: Send a SERR message, send a SCI message to the IICH, send a SMI message to the IICH, assert MCERR# on the front side bus, or do nothing. There is no determination of which errors go to which of the three error message schemes; it merely provides the capability for all combinations. It is the responsibility of the BIOS to determine the ultimate error reporting scheme. There will be an attempt to classify errors to whether they are fatal or non-fatal to more closely match the enterprise error presentation.

14.2.1 FERR/NERR Global Register Scheme

Figure 14-1. Global FERR/NERR Register Representation



The Global FERR register consists of three fields. The first or fatal field has 14b indicates the first signaled fatal global error from 14 different units. The second or non-fatal field indicates the first non-fatal global error that occurs from the same 14 different units. A non-fatal error may be either correctable or uncorrectable, but not

fatal. These two fields usually have at most one bit asserted in each field. In the event of simultaneous errors occurring in the same core clock, more than one bit in a field may be set. The third 4-bit field is reserved for future enhancements.

The Global NERR register consists of these same three fields with slightly different functionality. Instead of just the first fatal or non-fatal global errors recorded, this register indicates the second, third, fourth, etc. global errors that are reported by the IMCH.

These two registers do not indicate what the error was, they just indicate the severity of the error and what unit has more specific error information.

14.2.1.1 FERR/NERR Unit Registers

Each major unit will have a minimum of a pair of registers, known as the first error (FERR) and next error (NERR). Each unit has different and specific error bit definitions, and provides the specific type of error; information that is not found in the global registers. It is important to note that the unit FERR/NERR registers are simpler than the global for purposes of reuse and ease of implementation. While the global FERR register has a fatal and a non-fatal field, which lock down separately, the unit FERR register only has one field. The unit is however still required to send out separate fatal and non-fatal indications to the global FERR register if they detect both classifications of errors. Some units will support only one type. A unit that doesn't detect errors would not support either type.

14.2.1.2 Clearing FERR/NERR Registers

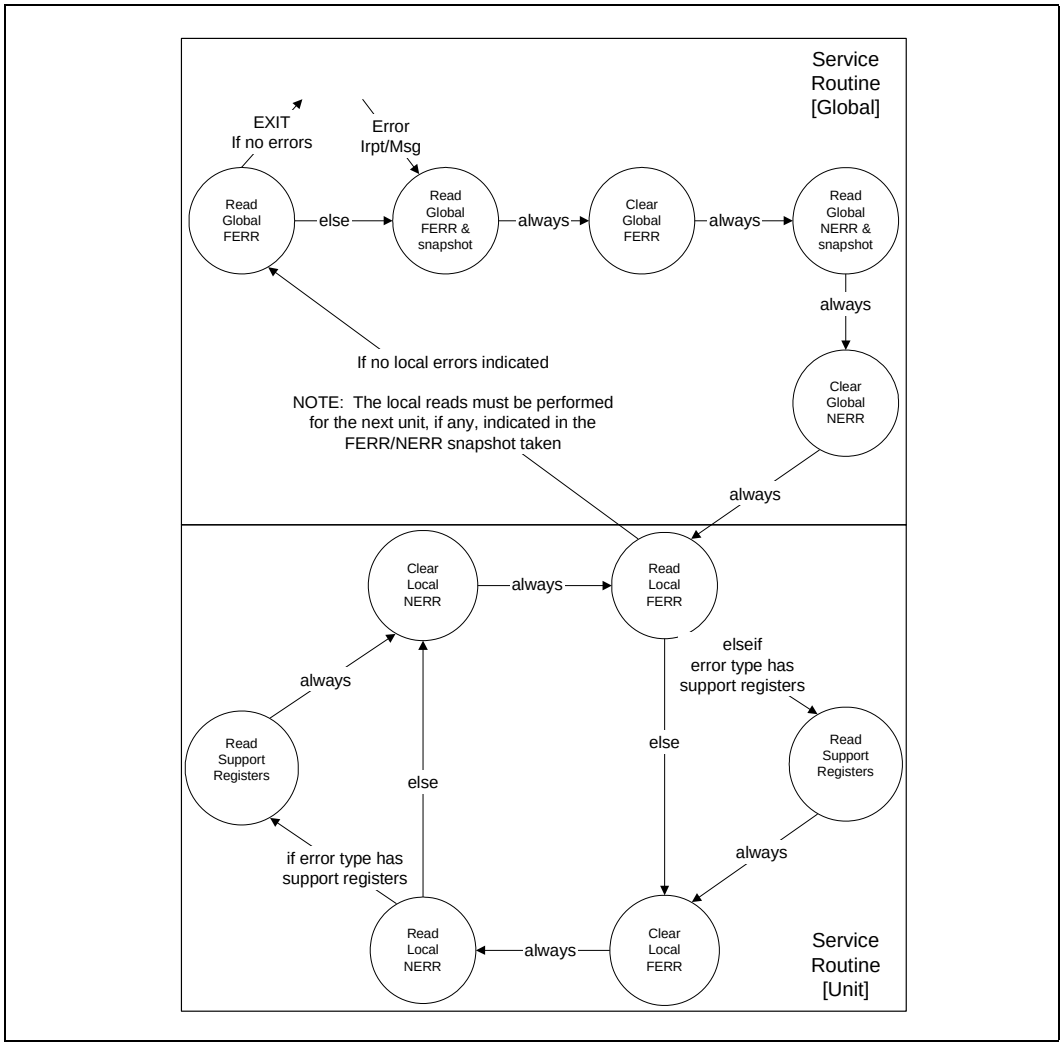
The following write-up is the recommended guideline to minimize the loss of errors information.

For a given FERR/NERR register pair, the FERR is read and then cleared first, and followed by the NERR. This sequence is true for either the global FERR/NERR register pair or any given unit. Any errors occurring after the FERR is cleared will then cause the FERR to have a non-zero value.

After the global FERR/NERR register pair is cleared, the unit FERR/NERR register pairs are interrogated, but only those indicated by the global FERR/NERR registers. Once the unit pair has been cleared, the unit FERR can be read again to ensure that no errors occurred during this local unit sequence. After the first unit FERR/NERR register pair has been serviced, this same sequence is performed for all other unit FERR/NERR register pairs that indicated errors in the global FERR/NERR registers. Once all unit error registers have been serviced, the final step is to read the global FERR register to determine if all system errors have been serviced. It is possible that errors could have occurred for a particular unit after that unit was serviced during the error routine, or that a unit had errors after the reading of the global FERR/NERR registers.

When clearing errors, software must clear all the FERR/NERR bits in the local interface registers before clearing the global FERR/NERR registers. If the local registers are not cleared first, then the global FERR/NERR registers will latch the same error again as soon as they are cleared. This implementation allows software to clear the local FERR/NERR registers, and then go clear the global FERR/NERR. S/W then reads back the global FERR/NERR and if it is non-zero, then a new error has occurred. If the global FERR/NERR has no bits set, then there are no more system errors.

Figure 14-2. FERR/NERR Service Routine



14.2.1.3 FERR/NERR Unit Specific

Each unit has different and specific error bit definitions.

14.2.1.4 SERR/SMI/SCI Enabling Registers

Each error reported has a full matrix of direction as to what error message it generates. For each unit FERR/NERR pair there are three more registers that enable each error for one of the three specific error messages. The logic does not appear to preclude the generation of all three messages for a single error, but this would not be a recommended configuration, and this needs to be looked into further. SERR stands for system error and is for reporting address and data parity errors, or any other catastrophic system error. SCI stands for system control interrupt and is a shareable interrupt used to notify the OS of ACPI events. SMI stands for System Management Interrupt and is an OS-transparent interrupt generated by events on legacy systems.

14.2.1.5 MCERR Enabling Registers

An additional entry to the matrix of error signaling paths is the MCERR (machine check error) enabling register. In addition to the SERR, SMI, and SCI enabling registers, the MCERR enabling register allows the occurrence of an error to result in the MCERR# signal to be asserted on the front side bus. Machine check error is asserted to indicate an unrecoverable error, which is not a bus protocol violation.

14.2.1.6 Error Escalation Register

Since all error bits in the error registers are fully configurable, meaning that a given error can be configured to go to any of the four messaging methods, no global error escalation mechanism is required. Although, the errors occurrence is accumulated in the global FERR/NERR registers, all error messaging is initiated from the units themselves, and not from a central location.

14.2.1.7 Error Masking

A new feature being added for CMI is the concept of an error masking register. Each unit has a mask register, which blocks the recognition/logging/reporting of each specific error type. Since the error will not be recognized when the corresponding mask bit is set, no error messages can be generated. This feature allows intelligent software to ignore specific error types during critical areas of code, where it does not want to be informed of errors that it will create, without ignoring other error types that it doesn't expect to happen. These mask bits will default to unmasked, and must be set by software or BIOS to take effect.

14.2.1.7.1 Locking DRAM Address and Syndrome on Errors

The first pair of error logging registers for CE (correctable errors) DRAM_SECF_ADD and DRAM_SECF_SYNDROME are locked when bit 0 of the DRAM_FERR is set. The second pair of error logging registers for CE (correctable errors) DRAM_SECN_ADD and DRAM_SECN_SYNDROME are locked when bit 0 of the DRAM_NERR is set. These pairs of two registers will retain their value even if new CE's are found. This allows the first (and possibly next) error to be captured and held instead of retaining the last. Corrected data errors as a result of either demand reads or scrubber-initiated traffic will be reflected in these error registers.

The logging register for UE (uncorrectable errors), DRAM_DED_ADD is locked when bit 1 of the DRAM_FERR or DRAM_NERR is set. This register holds the address of uncorrectable errors on data reads not initiated by the scrubber for either periodic or demand scrubbing.

The logging register for Scrub detected errors, DRAM_SCRUB_ADD should be locked when bit 2 of the DRAM_FERR or DRAM_NERR is set. This register holds the address for scrubber-initiated transactions for either demand or periodic memory scrubbing.

When the FERR/NERR registers are cleared the logging registers are free to update their contents until such time that either of these FERR/NERR registers again lock.

14.2.1.8 PCI Express Errors and Errors on Behalf of PCI Express

IMCH-specific error detection, masking, and escalation mechanisms operate on a parallel path to their standardized counterparts included in the *PCI Express* Interface Specification, Rev 1.0a*. PCI Express errors are classified as either correctable or uncorrectable. Uncorrectable errors are further broken down as fatal or non-fatal.

PCI Express specified correctable errors are logged in the Correctable Error Status Register (Device 2-3, Function 0, Offset 110 - 113h), unless they are masked by a corresponding bit in the Correctable Error Detect Mask Register (Device 2-3, Function 0, Offset 150-153h).

PCI Express specified uncorrectable errors are logged in the Uncorrectable Error Status Register (Device 2-3, Function 0, Offset 104-107h), unless they are masked by a corresponding bit in the Uncorrectable Error Detect Mask Register (Device 2-3, Function 0, Offset 14C-14Fh). The Uncorrectable Error Severity Register (Device 2-3, Function 0, Offset 10C - 10Fh) determines if bits in the Uncorrectable Status register are treated as uncorrectable fatal or uncorrectable non-fatal errors. The Device Status register (6Eh) bits are set when the corresponding category of bit is set in the uncorrectable and correctable status registers.

Reporting of non-masked error bits to the root complex hierarchy of PCI Express error registers is controlled on three different levels. Individual errors are masked for reporting by the Uncorrectable Error Mask (Device 2-3, Function 0, Offset 108-10Bh) and the Correctable Error Mask (Device 2-3, Function 0, Offset 114-117h) registers. Individual error category (fatal, non-fatal, correctable, or unsupported) reporting is enabled in the Device Control Register (Device 2-3, Function 0, Offset 6Ch) bits 3:0. Finally, uncorrectable error reporting (fatal or non-fatal) reporting may also be enabled by setting the SERR Enable bit in the PCI Command Register (Device 2-3, Function 0, Offset 04-05h).

There is an error pointer, in the Advanced Error Capability and Control Register (Device 2-3, Function 0, Offset 118-11Bh) which will log the first uncorrectable error that is enabled for reporting. Also some uncorrectable errors, when they are the first uncorrectable error, will log their corresponding header log in the Header Log Registers (Device 2-3, Function 0, Offset 11C-12Bh). An error pointer for unmasked correctable errors has been added in the Error Do Command Register (Device 2-3, Function 0, Offset 148-14Bh).

These internally detected errors when they are reported are referred to as virtual error messages. These are different from errors which are detected by the downstream device which then sends an error message to the root complex, which are referred to as externally detected or “received’ error messages. The received system error bit in the Secondary Status Register (Device 2-3, Function 0, Offset 1E-1F) is set when either fatal or non-fatal messages are received at the root complex.

At this point in the PCI Express error hierarchy, these virtual error messages are logically ORed with the received error messages, and will just be referred to as fatal, non-fatal, or correctable error messages, no reference to either virtual or received.

When enabled by the enable system error bit in the PCI Command Register, any fatal or non-fatal messages will set the signaled system error bit in the PCI Status register (Device 2-3, Function 0, Offset 06-07h). The Root Port Error Message Status Register (Device 2-3, Function 0, Offset 130-133h) will indicate first and multiple errors of each error message category, and the corresponding error source IDs of the first correctable and uncorrectable error messages will be the logged in the Error Source ID register (134h).

These errors that have been reported to the root complex can now be reported to the system, via the category enables in the Root Port Error Command Register (Device 2-3, Function 0, Offset 12C-12Fh) for interrupts. These interrupts can be in the form of

legacy type interrupts if so enabled in the PCI command register and MSI is not enabled, or message signaled interrupts if so enabled in the MSI Capabilities register (Device 2-3, Function 0, Offset 5A-5Bh).

The Root Port Control register (Device 2-3, Function 0, Offset 80-83h) enables errors to be reported to the system via other IMCH specific methods, again on a category basis. The Error Do Command register, selects between the four methods of system signaling, SERR, SCI, SMI, and MCERR.

The error model outside of PCI Express includes a local FERR/NERR pair of registers in each unit and a global FERR/NERR pair of registers that indicates which unit had problems. The Local FERR/NERR register pair (Device 2-3, Function 0, Offset 160-163h & 164-167h) includes PCI Express defined errors and additional detected errors within the PCI Express unit. This register pair has three sets of error bits for the three categories of errors: the first set for received messages, the second set for internally detected errors (virtual messages need not have been generated), and unit specific errors outside of the PCI Express spec, and the third set for device errors. This error scheme sets FERR/NERR error bits regardless whether or not they were reported via interrupt or other signaling method.

The signaling due to unit specific errors has its logic dependent on the PCI Express Unit Error Register (Device 2-3, Function 0, Offset 140-143h). The errors flagged in this register must be cleared before exiting the error service routine.

The signaling due to received messages has its logic dependent on the Root Error Message Status register (Device 2-3, Function 0, Offset 130-133h). The Root Error Status register must be cleared before exiting the error service routine.

The signaling due to internally detected PCI Express errors has its logic dependent on the Device Status register (Device 2-3, Function 0, Offset 6E-6Fh). The Device Status register must be cleared before exiting the error service routine.

Software must clear the global FERR first, and then the global NERR. Software then clears the local FERR register and the local NERR register of each unit in that order. After clearing FERR and then clearing NERR, the local FERR must be read to make sure that remains ‘0’ indicating no more errors have occurred during the clearing of these registers. After all units’ FERR & NERR registers have been cleared, the global FERR is again read to ensure that no additional errors occurred during the clearing sequence.

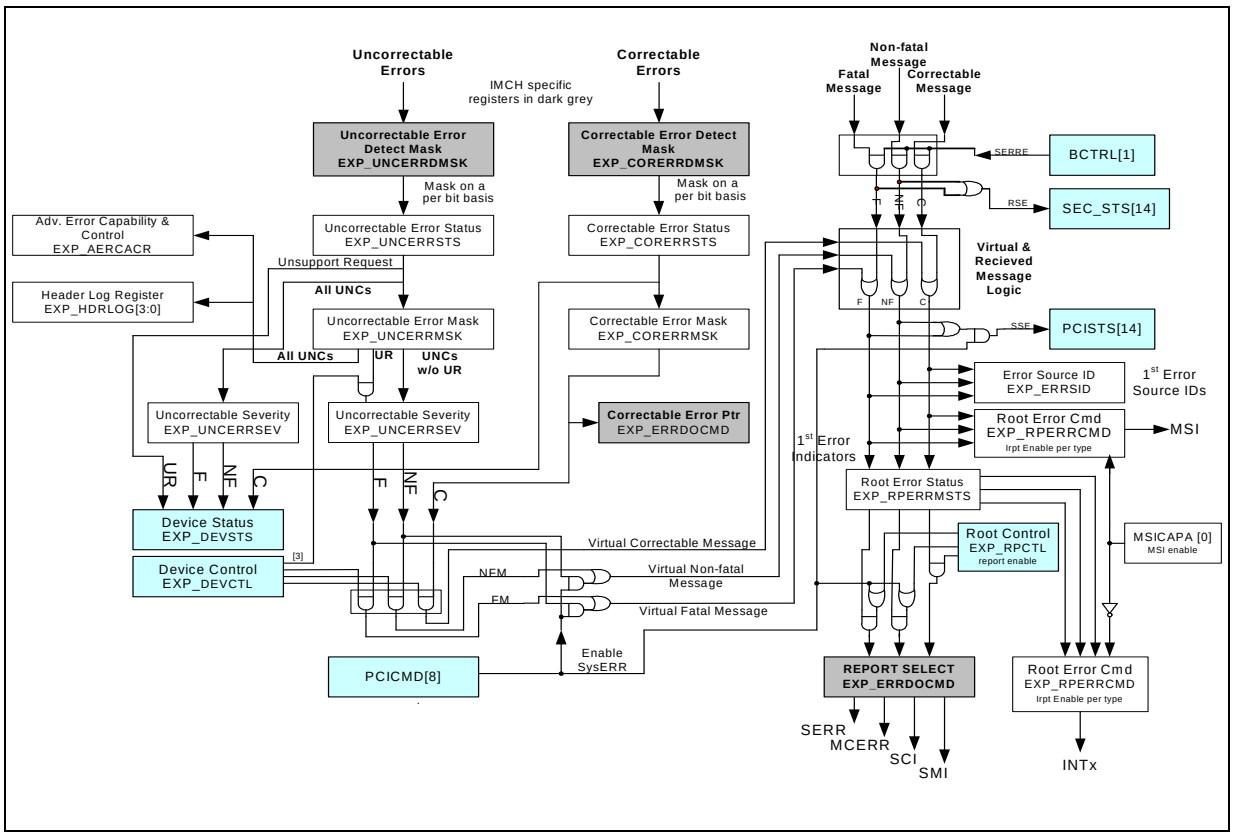
Since the PCI Express units have more hierarchy than other units, more registers must be cleared other than just the local FERR and NERR registers. After clearing the local FERR & NERR, one must also clear the Root Error Status, Unit Error Status, Device Status, Uncorrectable Error Status, and Correctable Error Status registers. One only needs to clear the PCI Status and Secondary Status registers if these are being utilized in a given particular error model. No logic depends on the state of any of these status bits. If not utilized, they can be ignored.

If a PCI Express error handler is used, with no knowledge of the FERR/NERR registers, then clear the PCI Express specific registers: Device Status, Uncorrectable Error Status, Correctable Error Status, and Root Error Status. The IMCH specific unit errors would not be enabled for reporting errors. Figure 14-3 helps to illustrate the relationship of the error registers from the *PCI Express* Specification.*

14.2.1.9 Configurable Error Containment at the Legacy Interface

Depending on the I/O devices in use, data errors could have catastrophic effects when allowed to propagate. The Legacy interface has the configurability of allowing the poisoning and propagation of data errors or to stop the data from transferring at all and escalate the data errors to the system. This is extreme behavior, which can be enabled or disabled, in order to prevent data corruption on a critical device, and is referred to as “stop and scream”. Refer to the Error Injection section for more details.

Figure 14-3. PCI Express Error Handling



14.3 Error Conditions Signaled

The IICH-notification action taken by the IMCH upon detection of an error is controlled through three registers. The SERRCMD register enables the generation of the SERR message, the SCICMD register enables the generation of the SCI message, and the SMICMD register enables the generation of SMI messages. Special cycle types of DO_SERR, DO_SCI, or DO_SMI may be transmitted to the IICH. The IICH receives the exception notification from the IMCH and may be configured to notify the processor of the condition.

Once the processor has been interrupted, it polls the system to determine the cause of the exception. If the IMCH initiated the exception condition by sending a message over NSI, then the processor is so informed by the IICH. At this point, the processor may read the IMCH's error status registers to determine the exact cause of the condition. The processor explicitly clears the status bit that points to the exception condition.

The IMCH in addition to signaling errors to the IICH for further handling, has added the capability of signaling the processor directly by use of the front side bus error signal MCERR#. The processor, upon observing this signal active, enters into special error handling code known as machine check code.

For each type of error detected in a given unit, there is a bit that corresponds to that error in the unit_FERR, unit_NERR, SERRCMD_unit, SMICMD_unit, SCICMD_unit, and MCERRCMD_unit registers. (Note that one and only one xCMD bit can be enabled per error type.) The first occurrence of an error type will be indicated by the bit assertion in the unit_FERR. If that error occurs again than the corresponding bit will be set in the unit_NERR register. When a bit is asserted in either the unit_FERR or unit_NERR, and if the corresponding enable bit is set in one of the named CMD registers, then an error signal will be asserted, corresponding to the name of the CMD register: DO_SERR, DO_SCI, DO_SMI, or DO_MCERR. The assertion of the DO_SERR signal also requires that the SERR enable in the PCICMD register is set. The assertion of the DO_SERR signal also causes the appropriate SERR signaled status bit to be set in the PCISTS register.

Table 14-1. Pseudocode for EDMA Errors (Sheet 1 of 2)

Condition	Source	Action	Status
The descriptor pointer in next descriptor address register is of incorrect type or range for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[7]=0 AND SERRCMD_EDMA[7]=1 AND R_EDGE{EDMA_FERR[31] OR EDMA_NERR[31]}); **DO_SMI** if EDMA_EMASK[7]=0 AND SMICMD_EDMA[7]=1 AND R_EDGE{EDMA_FERR[31] OR EDMA_NERR[31]}; **DO_SCI** if EDMA_EMASK[7]=0 AND SCICMD_EDMA[7]=1 AND R_EDGE{EDMA_FERR[31] OR EDMA_NERR[31]}; **DO_MCERR** if EDMA_EMASK[7]=0 AND **MC**ERRCMD_EDMA[7]=1 AND R_EDGE{EDMA_FERR[31] OR EDMA_NERR[31]};	EDMA_FERR[31]
The descriptor pointer in next descriptor address register is not aligned to eight double-word boundary for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[6]=0 AND SERRCMD_EDMA[6]=1 AND R_EDGE{EDMA_FERR[30] OR EDMA_NERR[30]}); **DO_SMI** if EDMA_EMASK[6]=0 AND SMICMD_EDMA[6]=1 AND R_EDGE{EDMA_FERR[30] OR EDMA_NERR[30]}; **DO_SCI** if EDMA_EMASK[6]=0 AND SCICMD_EDMA[6]=1 AND R_EDGE{EDMA_FERR[30] OR EDMA_NERR[30]}; **DO_MCERR** if EDMA_EMASK[6]=0 AND **MC**ERRCMD_EDMA[6]=1 AND R_EDGE{EDMA_FERR[30] OR EDMA_NERR[30]};	EDMA_FERR[30]
The source address does not comply with the source type or range for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[5]=0 AND SERRCMD_EDMA[5]=1 AND R_EDGE{EDMA_FERR[29] OR EDMA_NERR[29]}); **DO_SMI** if EDMA_EMASK[5]=0 AND SMICMD_EDMA[5]=1 AND R_EDGE{EDMA_FERR[29] OR EDMA_NERR[29]}; **DO_SCI** if EDMA_EMASK[5]=0 AND SCICMD_EDMA[5]=1 AND R_EDGE{EDMA_FERR[29] OR EDMA_NERR[29]}; **DO_MCERR** if EDMA_EMASK[5]=0 AND **MC**ERRCMD_EDMA[5]=1 AND R_EDGE{EDMA_FERR[29] OR EDMA_NERR[29]};	EDMA_NERR[29]
The source address is not aligned as specified by the source address bit for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[4]=0 AND SERRCMD_EDMA[4]=1 AND R_EDGE{EDMA_FERR[28] OR EDMA_NERR[28]}); **DO_SMI** if EDMA_EMASK[4]=0 AND SMICMD_EDMA[4]=1 AND R_EDGE{EDMA_FERR[28] OR EDMA_NERR[28]}; **DO_SCI** if EDMA_EMASK[4]=0 AND SCICMD_EDMA[4]=1 AND R_EDGE{EDMA_FERR[28] OR EDMA_NERR[28]}; **DO_MCERR** if EDMA_EMASK[4]=0 AND **MC**ERRCMD_EDMA[4]=1 AND R_EDGE{EDMA_FERR[28] OR EDMA_NERR[28]};	EDMA_FERR[28]
The destination address does not comply with the destination type or range for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[3]=0 AND SERRCMD_EDMA[3]=1 AND R_EDGE{EDMA_FERR[27] OR EDMA_NERR[27]}); **DO_SMI** if EDMA_EMASK[3]=0 AND SMICMD_EDMA[3]=1 AND R_EDGE{EDMA_FERR[27] OR EDMA_NERR[27]}; **DO_SCI** if EDMA_EMASK[3]=0 AND SCICMD_EDMA[3]=1 AND R_EDGE{EDMA_FERR[27] OR EDMA_NERR[27]}; **DO_MCERR** if EDMA_EMASK[3]=0 AND **MC**ERRCMD_EDMA[3]=1 AND R_EDGE{EDMA_FERR[27] OR EDMA_NERR[27]};	EDMA_FERR[27]

Table 14-1. Pseudocode for EDMA Errors (Sheet 2 of 2)

Condition	Source	Action	Status
The destination address is not aligned as specified by the destination address bit for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[2]=0 AND SERRCMD_EDMA[2]=1 AND R_EDGE{EDMA_FERR[26] OR EDMA_NERR[26]}); **DO_SMI** if EDMA_EMASK[2]=0 AND SMICMD_EDMA[2]=1 AND R_EDGE{EDMA_FERR[26] OR EDMA_NERR[26]}; **DO_SCI** if EDMA_EMASK[2]=0 AND SCICMD_EDMA[2]=1 AND R_EDGE{EDMA_FERR[26] OR EDMA_NERR[26]}; **DO_MCERR** if EDMA_EMASK[2]=0 AND **MC**ERRCMD_EDMA[2]=1 AND R_EDGE{EDMA_FERR[26] OR EDMA_NERR[26]};	EDMA_FERR[26]
Data parity Error in reading source data from system memory for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[1]=0 AND SERRCMD_EDMA[1]=1 AND R_EDGE{EDMA_FERR[25] OR EDMA_NERR[25]}); **DO_SMI** if EDMA_EMASK[1]=0 AND SMICMD_EDMA[1]=1 AND R_EDGE{EDMA_FERR[25] OR EDMA_NERR[25]}; **DO_SCI** if EDMA_EMASK[1]=0 AND SCICMD_EDMA[1]=1 AND R_EDGE{EDMA_FERR[25] OR EDMA_NERR[25]}; **DO_MCERR** if EDMA_EMASK1]=0 AND **MC**ERRCMD_EDMA1]=1 AND R_EDGE{EDMA_FERR[25] OR EDMA_NERR[25]};	EDMA_FERR[25]
Received configuration write command when EDMA is in Normal Mode for EDMA channel 3.	Internal	**DO_SERR** and set PCISTS10[SSE] if (PCICMD10[SERRE]=1 AND EDMA_EMASK[0]=0 AND SERRCMD_EDMA[0]=1 AND R_EDGE{EDMA_FERR[24] OR EDMA_NERR[24]}); **DO_SMI** if EDMA_EMASK[0]=0 AND SMICMD_EDMA[0]=1 AND R_EDGE{EDMA_FERR[24] OR EDMA_NERR[24]}; **DO_SCI** if EDMA_EMASK[0]=0 AND SCICMD_EDMA[0]=1 AND R_EDGE{EDMA_FERR[24] OR EDMA_NERR[24]}; **DO_MCERR** if EDMA_EMASK[0]=0 AND **MC**ERRCMD_EDMA[0]=1 AND R_EDGE{EDMA_FERR[24] OR EDMA_NERR[24]};	EDMA_FERR[24]
EDMA channel 2 errors	Internal	Same bit functionality as bits 31:24 except these are for EDMA channel 2. (Use bits 23:16)	
EDMA channel 1 errors	Internal	Same bit functionality as bits 31:24 except these are for EDMA channel 1. (Use bits 15:8)	
EDMA channel 0 errors	Internal	Same bit functionality as bits 31:24 except these are for EDMA channel 0. (Use bits 7:0)	

§ §

15.0 Platform Management (IMCH)

This chapter provides an overview of the system management support provided by the IMCH. There are two primary management support features in the IMCH:

- Integrated system management bus (SMBus) interface
- Architectural support for platform power management

Note: Material in this chapter is specific to the IMCH and does not apply to the IICH.

15.1 Integrated SMBus Interface

The IMCH provides a fully functional System Management Bus (SMBus) target interface, which provides direct access to all internal IMCH configuration register space. SMBus access is available to all internal configuration registers, regardless of whether or not the register in question is normally accessed via the memory-mapped mechanism or the standard configuration mechanism. This provides for highly flexible platform management architectures, particularly given a baseboard management controller (BMC) with an integrated network interface controller (NIC) function.

15.2 SMBus Target Architecture

The SMBus target integrated into the IMCH is compatible with the *System Management Bus (SMBus) Specification, Version 2*. A brief overview of the SMBus architecture is provided below for reference.

15.2.1 High Level Operation

The SMBus interface consists of two interface pins: a clock and serial data. Multiple initiator and target devices may be electrically present on the same pair of signals. Each target recognizes a start signaling semantic and recognizes its own seven-bit address to identify pertinent bus traffic. The IMCH address is hard-coded to 011_0000.

The protocol allows for traffic to stop in “midsentence,” requiring all targets to tolerate and properly “clean up” in the event of an access sequence that is abandoned by the initiator prior to normal completion. The IMCH is compliant with this requirement.

The protocol comprehends “wait states” on read and write operations, which the IMCH takes advantage of to keep the bus busy during internal configuration space accesses.

15.2.1.1 SMBus Register Summary

Table 15-1 provides a quick-reference summary of the SMBus target register space. These registers are part of the target itself and therefore not accessible by any other means other than the direct SMBus connection.

Table 15-1. SMBus Register Summary

Symbol	Full Name/Function
CMD	Command
BYTCNT	Byte Count
ADDR3	Bus Number (Only lower five bits are utilized)
ADDR2	Device/Function Number
ADDR1	Extended Reg Number (Bits 03:00 - 4 k page extension)
ADDR0	Register Number (offset into function space)
DATA3	Data, fourth byte (31:24)
DATA2	Data, third byte (23:16)
DATA1	Data, second byte (15:08)
DATA0	Data, first byte (07:00)
STS	Status, only for reads

Table 15-2. SMBus Memory-Mapped Register Summary

Symbol	Full Name/Function
CMD	Command
BYTCNT	Byte Count
ADDR3	Destination Memory (BAR Selection)
ADDR2	Address Offset 23:16 (Filler-used to zero out register)
ADDR1	Address Offset 15:08 (15:12 not used)
ADDR0	Address Offset 07:00 (11:00 used for 4 K page)
DATA3	Data, fourth byte (31:24)
DATA2	Data, third byte (23:16)
DATA1	Data, second byte (15:08)
DATA0	Data, first byte (07:00)
STS	Status, only for reads

Table 15-3. ADDR3 Memory Assignments

ADDR3	Destination Memory Assignments
00_000000	NSI
00_000001	EDMA
00_001000	DDR2
All others	Reserved

Table 15-1 and Table 15-2 indicate the sequence of data as it is presented on the SMBus following the byte address of the IMCH itself. This is not necessarily to indicate any specific register stack or array implemented in the IMCH. The registers can take on different meanings depending on whether it is a configuration or memory-mapped access type. The command indicates how to interpret the registers. Refer to the *System Management Bus (SMBus) Specification, Version 2* for interface protocol details.

15.2.1.2 Internal Register Access Mechanism

All SMBus accesses to the internal register space are initiated via a write to the CMD register. Any register writes received by the IMCH while a command is already in progress receive a NAK to prevent spurious operation. The master is no longer expected to poll the CMD register to prevent clobbering a command in progress prior to issuing further writes. The SMBus access is delayed by stretching the clock until such time that the data is delivered. Note that per the *System Management Bus (SMBus) Specification, Version 2*, this interval can not be longer than 25 ms. To set up an internal access, the four ADDR bytes are programmed, followed by a command indicator to execute a read or write. Depending on the type of access, these four bytes indicate either the Bus number, Device, Function, Extended Register Offset, and Register Offset; or the Memory-mapped region selected and the address within the region. The configuration type access utilizes the traditional bus number, device, function, and register offset; but also uses an extended register offset, which expands the addressable register space from 256B to 4 KB. The memory-mapped type access redefines these bytes to be a memory-mapped region selection byte and the memory address within the region. Table 15-2 and Table 15-3 show this information.

FSB-initiated accesses to registers are serviced through the configuration ring. For these registers, it is perfectly legal for an SMBus access to be requested while an FSB-initiated access is already in progress. The IMCH supports “wait your turn” arbitration to resolve all collisions and overlaps, such that the access that reaches the configuration ring arbiter first is serviced first while the conflicting access is held off. An absolute tie at the arbiter is resolved in favor of the FSB. Note that SMBus accesses must be allowed to proceed even if the internal transaction handling hardware and one or more of the other external interfaces are hung or otherwise unresponsive.

15.2.1.3 SMBus Register Definitions

15.2.1.3.1 CMD – Command Register

When written, this Command Register indicates the type and size of transfer. All configuration accesses from the SMBus port are initiated by writing to this register. While a command is in progress, all future writes or reads are NACK’ed by the IMCH to avoid overwriting registers while in use. The two command size fields allow for more flexibility on how the data payload is transferred, both internally and externally. The begin and end bits support the breaking of the transaction up into smaller transfers, by defining the start and finish of an overall transfer.

Table 15-4. Command (CMD) Register (Sheet 1 of 2)

Bit	Description
07	Begin Transaction Indicator 0 = Current transaction is NOT the first of a read or write sequence. 1 = Current transaction is the first of a read or write sequence. On a single transaction sequence this bit is set along with the End Transaction Indicator.
06	End Transition Indicator 0 = Current transaction is NOT the last of a read or write sequence. 1 = Current transaction is the last of a read or write sequence. On a single transaction sequence this bit is set along with the Begin Transaction Indicator.
05	Address Mode: Indicates whether memory or configuration space is being accessed in this SMBus sequence. 0 = Memory Mapped Mode 1 = Configuration Register Mode

Table 15-4. Command (CMD) Register (Sheet 2 of 2)

Bit	Description
04	Reserved - Set to 0.
03: 02	Internal Command Size: All accesses are naturally aligned to the access width. This field specifies the internal command to be issued by the SMBus slave logic to the IMCH core. 00 = Read Dword 01 = Write Byte 10 = Write Word 11 = Write Dword
01: 00	SMBus Command Size: This field specifies the SMBus command to be issued on the SMBus. This field is used as an indication of the length of the transfer so that the slave knows when to expect the PEC packet (if enabled). 00 = Byte 01 = Word 10 = Dword 11 = Reserved

15.2.1.3.2 BYTCNT – Byte Count Register

The byte count register indicates the number of bytes following the byte count register when performing a write or when setting up for a read. The byte count is also used when returning data to indicate the following number of bytes (including the status byte), which are returned prior to the data. Note that the byte count is only transmitted for block type accesses on SMBus. SMBus word or byte accesses do not use the byte count.

Table 15-5. Byte Count Register

Position	Description
07:00	Byte Count: Number of bytes following the byte count for a transaction.

15.2.1.3.3 ADDR3 – Address Byte 3 Register

This register must be programmed with the Bus Number of the desired configuration register in the lower five bits for a configuration access. For a memory-mapped access this field selects which memory-map region is being accessed. There is no status bit to poll to see if a transfer is currently in progress, because by definition, if the transfer completed, the task is done. The clock stretch is used to guarantee the transfer is truly complete.

The EP80579 does not support access to other logical bus numbers via the SMBus port. All registers “attached” to the configuration mechanism that the SMBus has access to, reside on logical bus#0.

Table 15-6. Address Byte 3 Register

Position	Configuration Register Mode Description	Memory Mapped Mode Description
07:05	Ignored.	Memory map region to access. 00h = NSI 01h = EDMA 08h = DDR2 Others = Reserved
04:00	Bus Number: Must be zero: the SMBus port can only access devices on the IMCH and all devices are bus zero.	

15.2.1.3.4 ADDR2 – Address Byte 2 Register

This register must be programmed with the Device Number and Function Number of the desired configuration register if for a configuration type access, otherwise it must be set to zero.

Table 15-7. ADDR2 – Address Byte 2 Register

Position	Configuration Register Mode Description	Memory Mapped Mode Description
07:03	Device Number. Can only be devices on the IMCH.	Zeros used for padding.
02:00	Function Number.	

15.2.1.3.5 ADDR1 – Address Byte 1 Register

This register must be programmed with the upper address bits for the register with the 4K region. Whether it is a configuration or memory-map type of access, only the lower bits are utilized, the upper four bits are ignored.

Table 15-8. ADDR1 – Address Byte 1 Register

Position	Description
07:04	Ignored.
03:00	Extended Register Number. Upper address bits for the 4 K region of register offset.

15.2.1.3.6 ADDR0 – Address Byte 0 Register

This register indicates the lower eight address bits for the register within the 4 K region, regardless of whether it is a configuration or memory-map type of access.

Table 15-9. ADDR0 – Address Byte 0 Register

Position	Description
07:00	Register Offset.

15.2.1.3.7 DATA – Data Register

This field is used to receive the read data or to provide the write data associated with the desired register.

At the completion of a read command, this field contains the data retrieved from the selected register. All reads return an entire aligned Dword (32 bits) of data.

The appropriate number of byte(s) of this 32-bit logical register must be written with the desired write data prior to issuing a write command. For a byte write only, bits 7:0 are used, for a Word write, only bits 15:0 are used, and for a Dword write, all 32 bits are used.

Table 15-10. Offset 04-07: DATA - Data Register

Bits	Type	Reset	Description
31:24	RW	00h	Byte 3 (DATA3): Data bits [31:24]
23:16	RW	00h	Byte 2 (DATA2): Data bits [23:16]
15:08	RW	00h	Byte 1 (DATA1): Data bits [15:8]
07:00	RW	00h	Byte 0 (DATA0): Data bits [7:0]

15.2.1.3.8 STS – Status Register

For a read cycle, the data is preceded by a byte of status. Table 15-11 shows how these bits are defined.

Table 15-11. Status Register

Position	Description
07	Internal Timeout 0 = SMBus request is completed within 2 ms internally 1 = SMBus request is not completed in 2 ms internally
06	Ignored
05	Internal Master Abort 0 = No Internal Master Abort Detected 1 = Detected an Internal Master Abort
04	Internal Target Abort 0 = No Internal Target Abort Detected 1 = Detected an Internal Target Abort
03:01	Ignored
00	Successful 0 = The last SMBus transaction was not completed successfully 1 = The last SMBus transaction was completed successfully

15.2.1.4 Unsupported Access Addresses

It is possible for an SMBus master to program an unsupported bit combination into the ADDR registers. The IMCH does not support such usage, and may not gracefully terminate such accesses.

15.2.1.5 SMBus Transaction Pictograms

Since the new SMBus target interface is of enterprise origin, it is more complex than the original SMBus target interface of desktop origin. The following drawings are included to demonstrate the different types of transactions, especially how they can be broken up into multiple smaller transfers.

Figure 15-1. Dword Configuration Read Protocol

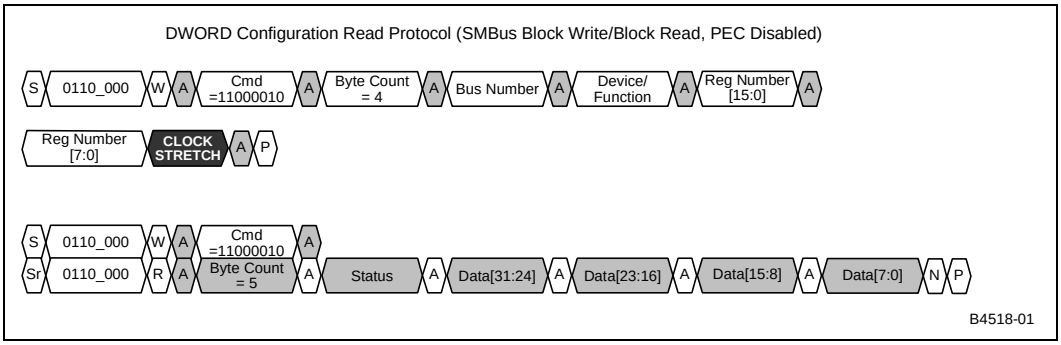


Figure 15-2. Dword Configuration Write Protocol

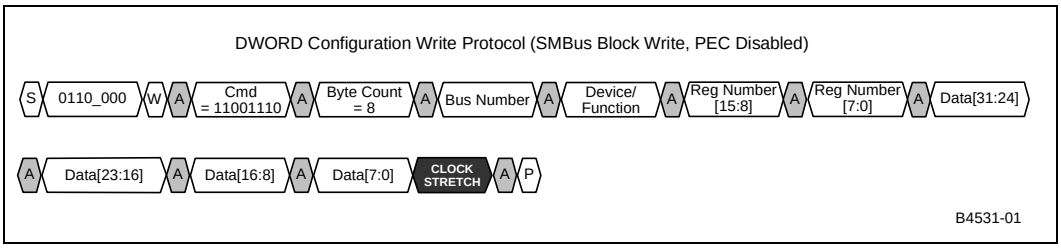


Figure 15-3. Dword Memory Read Protocol

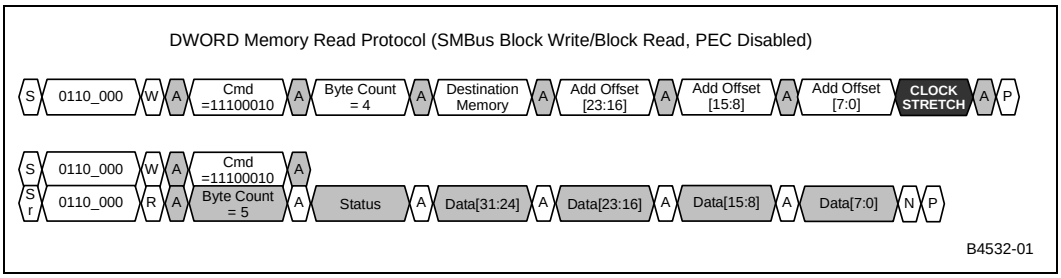


Figure 15-4. Dword Memory Write Protocol

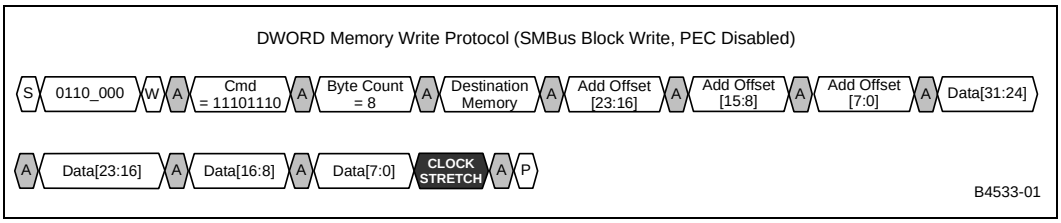


Figure 15-5. Dword Configuration Read Protocol

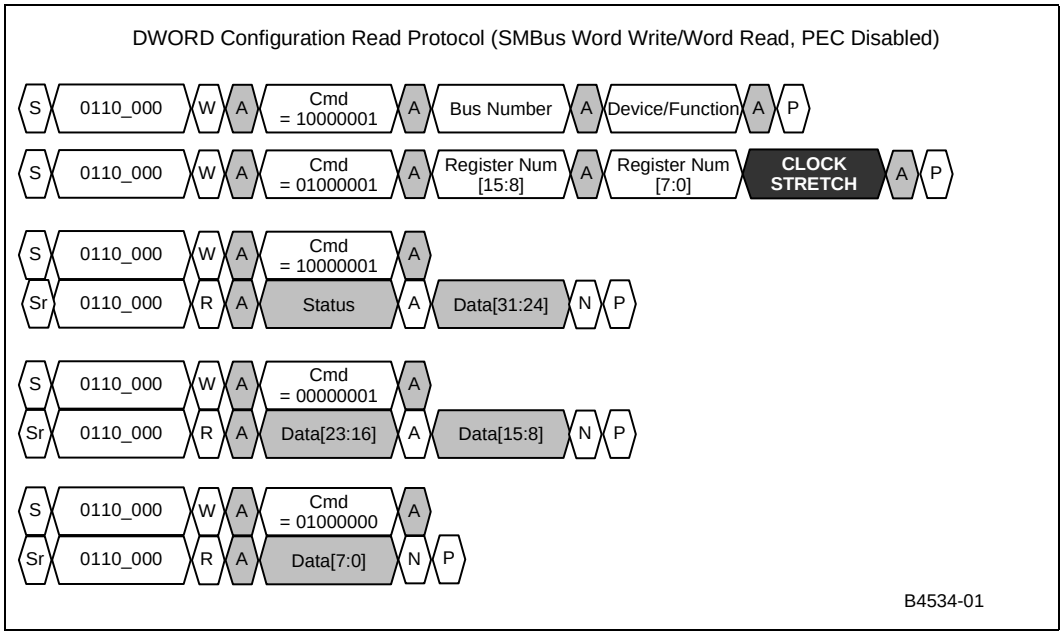


Figure 15-6. Dword Configuration Write Protocol

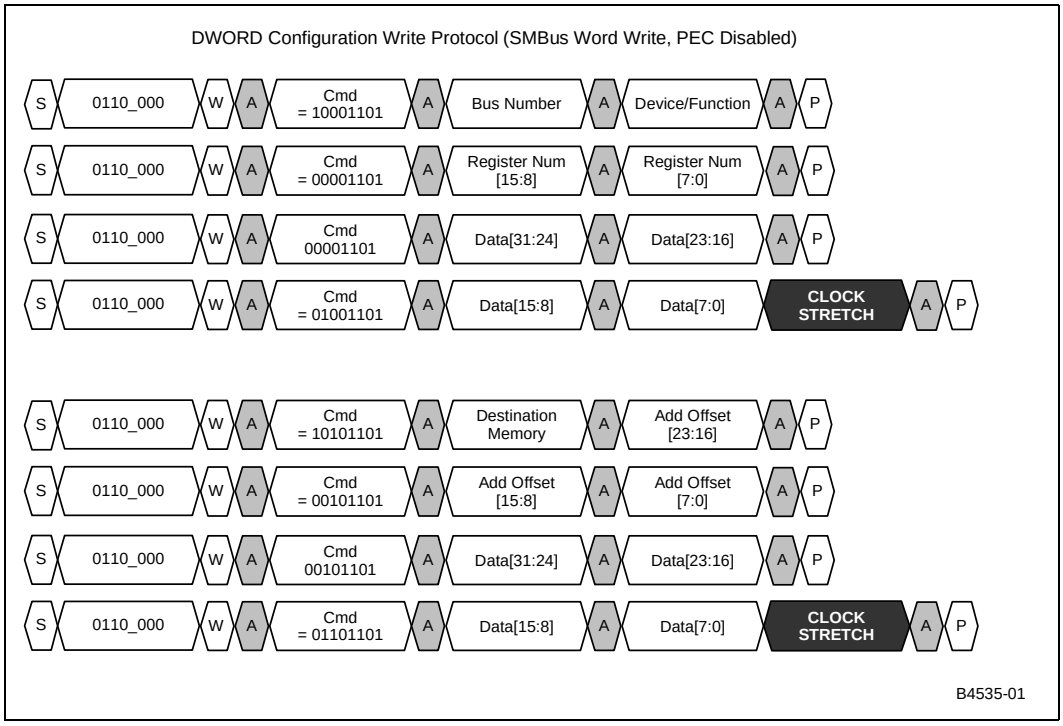


Figure 15-7. Dword Memory Read Protocol

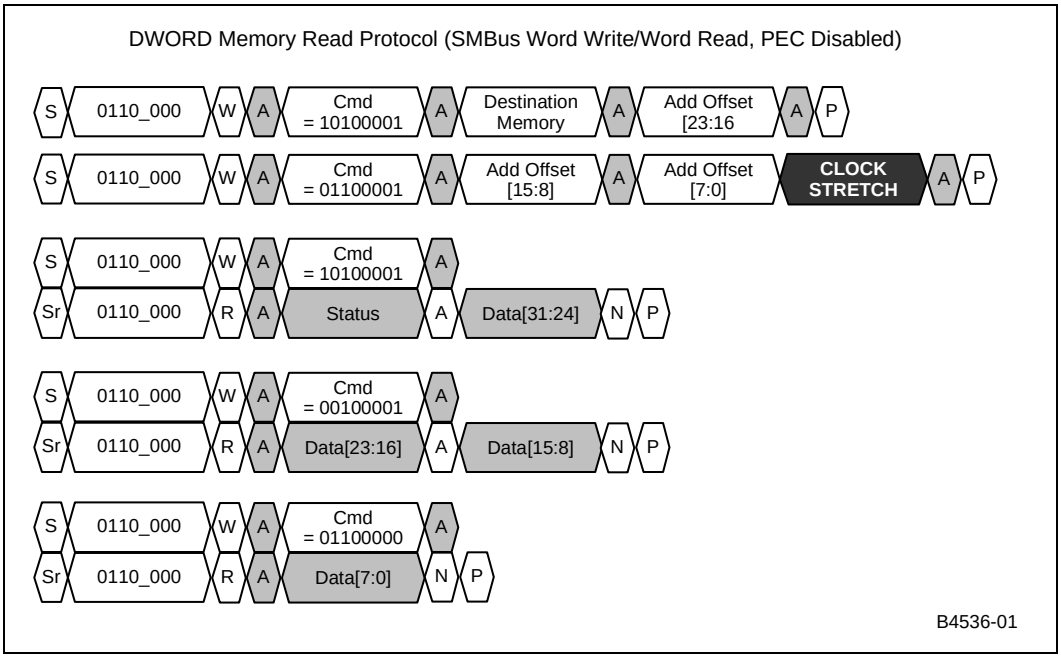
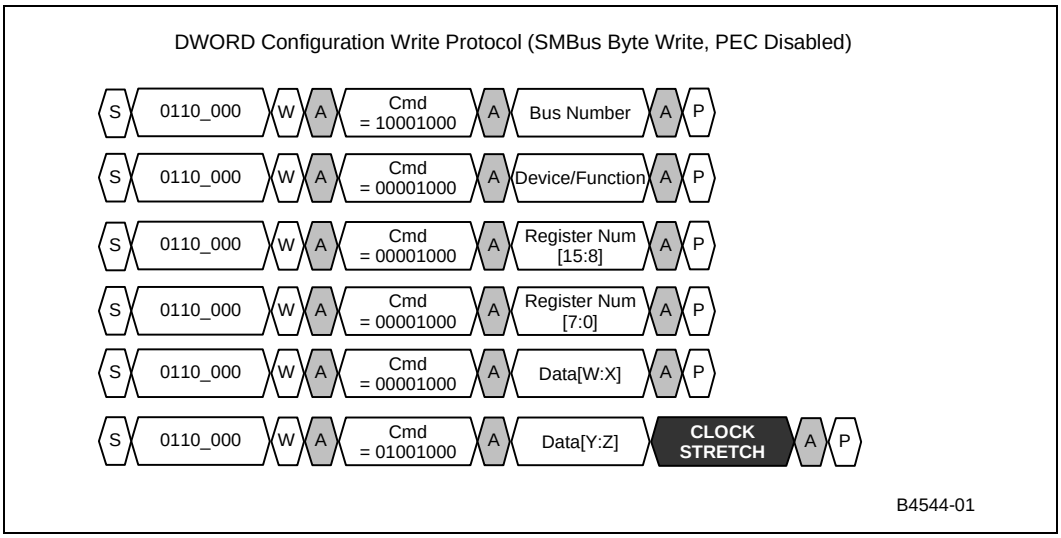


Figure 15-8. Dword Configuration Write Protocol



15.2.2 Suggested SMBus Usage Models

15.2.2.1 Remote Error Handling

The EP80579 supports error escalation via both SMI and MCERR FSB signaling, thus error handling may be implemented in system management mode (SMM) software, machine check architecture (MCA) software, or a combination of the two. Such software could direct a BMC with an integrated NIC to “call home” when errors are reported by the IMCH. The BMC could then interrogate internal IMCH error logging registers under remote control across the network interface, providing full identification and isolation of reported errors as described elsewhere in this document. The possibility also exists for remotely managed reconfiguration via the SMBus target port, as well as remotely managed system reboot via the BMC (if necessary).

15.2.2.2 Remote Platform Monitoring

The SMBus target also provides a sophisticated BMC with the capability to monitor the health of an EP80579-based platform, such that statistics on correctable error location and frequency may be tracked remotely in an effort to anticipate and prevent more serious failures.

The IMCH includes significant RASUM functionality on both its memory subsystem and its PCI Express interfaces. Some types of errors are expected at a modest frequency within a platform of this complexity, and the IMCH provides internal hardware to track the frequency of such errors. These include correctable ECC errors on the memory interface, as well as transient communication errors on the high-speed serial PCI Express interfaces – refer to Chapter 14.0, “RAS Features and Exception Handling” for further details.

The BMC could be remotely directed to periodically poll the internal error logging registers of the IMCH, permitting a remote management software package to maintain a running profile of error types and frequencies experienced by an EP80579-based platform. Changes in error frequency or type could be flagged by the remote monitoring software to prompt follow-up preventative maintenance on the platform.

15.3 Platform Power Management Support

The IMCH is compatible with the *PCI Bus Power Management Interface Specification, Revision 1.1* (referred to here as PCI-to-PMI). The IMCH is also compatible with the *Advanced Configuration and Power Interface Specification, Rev. 2.0 (ACPI)*. The EP80579 is designed to operate seamlessly with operating systems employing these specifications.

The anticipated implementation for platform power management control is an add-on component connected to the IICH component of the core logic via its LPC and/or SMBus interfaces.

15.3.1 Supported System Power States

The IMCH and the system power states are analogous, thus no “device” power states are defined for the IMCH. As a result, the IMCH power state may be directly inferred from the system power state.

Like all systems, EP80579-based platforms must support the S0 (fully active) state at a minimum. The IMCH also supports S1 (Idle), S3 cold (suspend to RAM), S4 (suspend to disk), and S5 (soft off).

S2 (power-on suspend) and S3 hot are not supported.

S3 support requires specialized internal hardware. A request to enter the S3 power state is communicated to the IMCH by the IICH. In response, the IMCH flushes all data from the internal coherent write buffer, sequences all active DIMM rows into their “self-refresh” state, and then returns an Ack_Sx special cycle to the IICH. Upon completion of this sequence, the IMCH tolerates the removal of all clock references and power sources, save the DDR2 interface power. DDR2 interface power must be supplied so that the IMCH may hold the DIMMs in self-refresh. A full system initialization and configuration sequence is required upon system exit from the S3 state, as all (non-AUX) internal configuration information has been throughout the platform, but exit latency is much lower than it would be from S5, as the memory image has been maintained.

Note: The Go_S3 message indicates that the IICH is getting ready to put the system into S3, S4 or S5 state.

The extra internal logic support for S3 and S4 is not required for the S5 (soft off) system power state because all data in the memory array is lost regardless, and the coherent write buffer is architecturally part of the data stored in main memory.

15.3.1.1 Supported CPU Power States

EP80579-based platforms support the C0, C1, C2 and C3 states as defined by the *Advanced Configuration and Power Interface Specification (ACPI)*. This implies that the core logic anchored by the IMCH properly understands and handles messaging between the IMCH and the FSB to facilitate transitions into and out of these states.

15.3.1.2 Supported Device Power States

The IMCH supports all PCI-to-PMI and PCI Express messaging required to place any subordinate device on any of its PCI Express ports into any of the defined device low power states. Peripherals attached to the PCI segments provided via a PXH component may be placed in any of their supported low power states via messaging directed from the IMCH through the intervening PCI Express hierarchy. Directly attached native PCI Express devices are not limited in their available low power states, although not all available states support the downstream device “wake-up” semantic.

Further details about PCI Express power management support and accompanying PCI Express and subordinate device power management support are provided in Section 15.3.3, “PCI Express Interface Power Management” on page 382.

15.3.1.3 Supported Bus Power States

No low power bus states are supported by the EP80579 on its internal NSI interface between the IMCH and the IICH. Also, IMCH does not support placement of the IICH only into any low-power state below D0, other than as a side-effect of placing the entire system into one of the S3 cold, S4, or S5 states.

Significant low power mode support is provided for the several IMCH PCI Express ports, as detailed in Section 15.3.3, “PCI Express Interface Power Management” on page 382.

15.3.2 DDR2 Interface Power Management

DDR2 self-refresh is supported as an integral piece of the S3 support.

15.3.3 PCI Express Interface Power Management

In PCI Express, the traditional bus (B*) power states assigned to system buses are replaced by link (L*) power states, which are largely managed by hardware without software intervention. Entry into and out of these states may be initiated by two distinct mechanisms:

- traditional PCI-PMI type software managed state changes
- non-traditional PCI Express autonomous hardware state changes

The latter transition type is designated “Active State Power Management,” (ASPM) and included with the *PCI Express Interface Specification, Rev. 1.0a*.

15.3.3.1 PCI Express Link Power State Definitions

Support for all of the following PCI Express link power states is required for *PCI Express Specification* compatibility:

Note: The IMCH does not support all PCI Express link power states.

- L0 – Active state with all operations enabled (default state after platform initialization).
- L0s –Low latency, energy saving standby state, disabling exchange of both transaction layer packets and device link layer messages. This state is used exclusively by the ASPM PCI Express function, with entry and exit managed autonomously by PCI Express interface hardware.
- L1 – (Only supported in software managed state changes) Moderate to high latency, very low power, standby state, disabling exchange of both transaction layer packets and device link layer messages. Entered when the downstream device is programmed to a device power state below the D0 active state, or optionally under hardware control during ASPM. The clock remains active in L1, and exit from this state may be initiated by either the upstream or the downstream device.
- L2/L3 Ready – Staging point for removal of main power and clocking. New intermediate state not directly related to PCI PM D-state transitions, nor to ASPM. Hand-shaking lands the link in this state in anticipation of power removal, at which point the link moves to either L2 or L3 depending upon the presence of Vaux.
- L2 – High latency, very low deep sleep state, disabling exchange of transaction layer packets and device link layer messages. L2 is characterized by removal of clocking and main power, but presence of Vaux power. Exit is initiated by restoring clocking and power, and full initialization.
- L3 – High latency, link off state with power, Vaux, and clock reference removed. Exit is initiated by restoring clocking and power, and full initialization.

The IMCH is fully compliant with the *PCI Express Specification*, but does not support the optional L1 state via the ASPM mechanism. Refer to the *PCI Express* Interface Specification, Rev. 1.0a* for further detail on the link states and specific information on entry and exit mechanisms.

15.3.3.2 Software Controlled PCI Express Link States

Software managed device power state changes do not explicitly control the power L-state of PCI Express links. Instead the L-state is inferred by hardware from the PCI-PMI power state of the devices attached to that link. When PM software transitions a PCI Express device to a low power state, that device automatically negotiates with hardware to bring its upstream link into the appropriate link power state.

No link is allowed to be in a link power state “below” that which is dictated by its attached components.

Table 15-12 defines the legal relationships between link and attached device power states.

Table 15-12. Relationship Between Link and Device PM States

Downstream Component D-State	Permissible Upstream Component D-State	Permissible Interconnect L-State
D0	D0	L0, L0s, L1†
D1	D0, D1	L1
D2	D0, D1, D2	L1
$D3_{hot}$	D0, D1, D2, $D3_{hot}$	L1, L2/L3 Ready
$D3_{cold}$	D0, D1, D2, $D3_{hot}$, $D3_{cold}$	L2, L3

† Entry into L0s or L1 while attached devices remain in D0 only occurs as a part of ASPM. Per the PCI Express spec, L0s support is mandatory, while L1 is optional. **L1 is not supported by ASPM**.

Several new semantics are introduced with PCI Express to support PCI-PMI compatible software managed device and link power state transitions. The majority of the new functionality is to accommodate an essentially edge-triggered, in-band message scheme supporting multi chassis cabled system topologies, which must replace the function of traditional level-sensitive board traces for PM event and wake signaling. Further details about PME signaling appears in Section 15.3.5, “PME Support” on page 385.

The IMCH supports messaging to facilitate transition of attached PCI Express devices to power states D0, D1, D2, and D3 (both $D3_{HOT}$ and $D3_{COLD}$). All attached devices are required by the *PCI Express Specification* to support the D0 and both D3 states, while D1 and D2 support are optional. Software should confirm device support of the optional D1 and D2 states prior to attempting their use on any attached PCI Express device.

In the D1, D2, and $D3_{HOT}$ states, the attached device is required to suppress initiation of any link traffic other than PME initiation (if enabled) as a master, and must only accept configuration transactions as a target. Functional context is maintained in the D1 and D2 states, such that full initialization of the attached device is not required upon the wake-up transition back to the D0 state. In both D3 states, functional context is not maintained, and full initialization is required after a transition back to D0.

Placing an attached device into a low power state results in automatic transition of the associated PCI Express port to its L1, L2 or L3 link state (depending upon the device power state). To save additional power in the L2 state, the platform power manager must remove the reference clock from the link. The IMCH does not provide the necessary internal clock generation and distribution control to allow clock removal from one PCI Express port interface without impacting the operation of its peer ports on the IMCH. CMI does not provide support on its PCI Express link interfaces for the in-band “tone” required to wake from such a state.

15.3.3.3 Hardware Controlled PCI Express Link States

Active-state power management (ASPM) support is a required component of *PCI Express Specification* compliance and is intended to provide granularity and flexibility for PCI Express components to dynamically manage their own power consumption without software supervision and support. All native PCI Express components, regardless of device class, must support active transitions to and from the L0s low power link state as a minimum and may optionally support active transitions to and from the L1 low power link state.

The IMCH does not support ASPM entry into the L1 state under hardware control. CMI negatively acknowledges (NAK) ASPM requests for L1 state transitions. IMCH configuration registers reflect this level of support for ASPM.

All ASPM functionality is disabled by default upon system power-up. It is the responsibility of software to verify a viable platform clocking configuration prior to enabling ASPM functionality within the IMCH or in any attached PCI Express devices. In topologies where independent clock references are used at any point within the PCI Express subsystem hierarchy, the “fast training” sequence associated with ASPM is not guaranteed to successfully revive the associated link, and ASPM must remain disabled in the devices at both ends of that link.

15.3.3.4 System Clocking Solution Dependencies

The topology of the platform clocking solution dictates the viability of ASPM on each of the PCI Express links. This is because the nature of the clocks directly impacts the amount of time required to re-acquire bit and symbol lock in the receiver after an arbitrarily long non-communicative period. When both ends of a link share a clock source, they “wander” together over the period they are out of communication with each other, and accordingly require a relatively brief period of training to re-acquire lock. When the two ends of a link utilize completely independent clock references, they may become arbitrarily out of phase with each other while they are in low power states, and therefore require a significantly longer amount of time to re-acquire lock upon waking. For this reason, the *PCI Express Interface Specification* provides for software discovery and communication of the actual clocking topology within the system prior to enabling the ASPM feature on any link within the system.

There are two primary components to the clocking discovery mechanism. First, all downstream ports, such as those on the IMCH root device, must report whether they use the same clock source as that provided to the slot (or down-device) connected to that port in the platform. This information is recorded in the Slot Clock Configuration bit of the Link Status Register for each port and system BIOS is required to initialize these bits accordingly. Second, all add-in devices must report whether they utilize the clock reference provided on the add-in slot via the same bit in the same register of their capability structure.

System software may examine the settings of the Slot Clock Configuration bits of both the upstream and downstream devices for each port in the system, and determine whether a common clock reference is in use. This information is then communicated to both the upstream and the downstream devices via programming of the Common Clock Configuration bit of the Link Status Register. The setting of this bit determines the reported exit latency requirements for the L1 state. System software may then compare the exit latency requirements with the tolerated exit latencies of the attached device and determine whether or not to enable ASPM for each link the system. All ASPM functionality defaults to disabled at power-on and remain so unless system software determines it may be enabled.

The “N_FTS” parameters exchanged during initial training corresponds to the “long” exit latencies associated with independent clocks. If software later sets the Common Clock Configuration bits, it is also necessary to force link retraining in order to update the exchanged N_FTS information.

15.3.3.5 Device and Link PM Initialization

All PCI Express devices power-on into the $D0_{uninitialized}$ state and remain in that non-communicative state until they have been configured and at least one of the Memory Space Enable, I/O Space Enable, or Bus Master Enable bits has been set by system software, at which point the device automatically transitions to the $D0_{ACTIVE}$ state indicative of normal operation.

15.3.4 Device and Slot Power Limits

All add-in devices must power-on to a state in which they limit their total power dissipation to a default maximum according to their form-factor (10 W for add-in edge-connected cards). When BIOS updates the slot power limit register of the root ports within the IMCH, the IMCH automatically transmits a Set_Slot_Power_Limit message with corresponding information to the attached device.The platform BIOS is responsible for properly configuring the slot power limit registers in the IMCH. Failure to do so may result in attached endpoints remaining completely disabled in order to comply with the default power limitations associated with their form-factors.

15.3.5 PME Support

All information in this section refers to the IMCH PME support. See Chapter 27.0, "Power Management" for PME support in IICH.

In CMI systems, only the system power manager or a device within the PCI Express hierarchy may initiate a power state change. Thus the only Power Management Event (PME) signaling support required in the IMCH is that associated with PCI Express. Note that a device bridging to another technology, such as a PXH bridging to PCI-X, may convert traditional PME signaling into PCI Express in-band PME messaging and thereby meet this requirement of the IMCH.

PME signaling in PCI Express is crafted to accomplish two distinct functions. First, it provides a signaling mechanism for devices requiring service to propagate a wake-up request to the power management controller. Second, it provides a messaging mechanism for devices requesting a power state change to pass their unique location within the PCI Express hierarchy to the power management controller. The combination of these two functions provides great flexibility and controllability for the power manager.

15.3.5.1 PME Wake Signaling

Wake signaling is only required to provide for device-initiated transition out of low power states where clock and/or power have been removed from the sleeping device. The PME mechanism does not require a wake-up function for attached devices still powered and receiving an interface reference clock, as devices in this state may simply initiate PME messaging directly. Wake is only required if the device wishing to initiate a PME message cannot do so without first requesting a change to the system clocking and power profile from the power management controller.

The wake signaling aspect of the system power management solution may vary in elegance and granularity. Depending upon the support level provided by the power management controller, a wake-up request from any given device may cause power and clocking to be restored to the entire system, to just the affected branch of the PCI Express hierarchy, or only to the requesting device.

While the *PCI Express Interface Specification* provides for two distinct wake signaling mechanisms, CMI supports only the legacy mechanism described below.

15.3.5.1.1 Legacy Wake Mechanism

The legacy wake signaling mechanism is analogous to that used in historical PCI-based system designs. In this case, the platform architect is responsible for crafting paths routing collected wake signals between wake-capable devices and the management controller without participation from the IMCH and IICH equivalent devices. The collection of wake logic must run on auxiliary power, and must comprehend the potential for devices both with and without supplied auxiliary power co-existing on the same branch of the PCI Express hierarchy. Refer to the *PCI Express Interface Specification* for further details on legacy wake signaling.

While familiar and well understood, this mechanism does not provide for device-initiated wake-up in the fully A/C coupled implementation of a multi-chassis PCI Express-based system solution. The limitation imposed upon CMI-based system is that remote-chassis devices must not be placed in a low-power state with the PCI Express link clocking and/or power disabled if legacy style wake signaling is desired for any peripheral in that remote chassis. It is still possible for software to place peripheral devices in low power sleep states and to manage the device state of the PCI Express device attached to the inter-chassis cable. Devices in the remote chassis may still initiate power state changes via PME messaging provided the inter-chassis link has not been placed into an uncommunicative state. An alternative for the platform architect would be to place a compatible switch device between the IMCH and the remote chassis that supports the in-band wake mechanism described below and rely on the switch to forward wake events to the management controller.

The platform architect should ensure that the power management controller can adequately isolate the source of a PME wake request as required to take appropriate power management
wake-up action.

15.3.5.2 PME Messaging

Once the link requesting a power state change has a communicative upstream link, it sends the PM_PME packet upstream toward the root device (IMCH), which in turn is responsible for notifying the management controller. This constitutes an in-band "virtual wire" signaling mechanism to replace the historical solution that involved multiple independent board traces routing PME requests to the power manager. Because the PM_PME propagates "in-band" on the PCI Express interface without any side-band signaling support, PME functionality is made available to multi-chassis system solutions.

The IMCH collects and "OR" PME requests from all logical PCI Express ports and propagates it to IICH over the NSI link as an Assert_PMEGPE message. The IICH then generates a specified interrupt to wake the power manager and invokes power management software. The interrupt service routine may then interrogate the various PM status registers to determine the source(s) of PME. The IMCH would send a Deassert_PMEGPE message over NSI link after the power state change request has been serviced.

15.3.6 BIOS Support for PCI Express PM Messaging

The *PCI Express Specification* stipulates hierarchical messaging semantics enforced by the root device (IMCH) to guarantee proper entry into and exit from
unpowered device states. CMI ACPI BIOS must make special allowances for support of these semantics. There are two sets of messages that must be software-assisted in CMI based platforms to support power-off device states within the PCI Express hierarchy.

15.3.6.1 PCI Express PME_TURN_OFF Semantic

Prior to removing power from any attached PCI Express links anywhere in the hierarchy, the root device must broadcast a PCI Express "PME_TURN_OFF" message to all downstream devices on the affected PCI Express port. The receiving devices propagate this message to all subordinate PCI Express ports (if any), collect "PME_TO_ACK" acknowledgement packets, and finally return a "PME_TO_ACK" transaction layer packet back to the root device. Once all active ports have acknowledged, the power management device may be notified that it is cleared to modify the collective power state of the PCI Express hierarchy. These message packets have posted semantics on the interface, thus the turn-off "pushes" all prior packets to

their endpoints, and the acknowledge “pushes” any pending inbound traffic all the way to the root. This prevents “trapping” transactions or PME messages somewhere in the hierarchy at the time power is dropped, ultimately causing them to be lost.

In a pure PCI Express design, the PME_TURN_OFF packet would originate directly at the power manager, or perhaps at the IICH providing connection between the power manager and the remainder of the core logic. Neither the power manager nor the IICH is aware of the PCI Express messaging mechanism, thus the IMCH provides device-specific control and status bits for use by its ACPI BIOS.

The sequence of events to place a PCI Express device in an unpowered state is as follows:

1. PCI-PM or ACPI-compliant O/S software is called to place the system into a low-power sleep state (S3, S4 or S5), prepares for suspension, and calls ACPI BIOS to carry out the platform power transition.
2. The BIOS then communicates to the root complex that all PCI Express devices must prepare for power-off. This is accomplished through the device-specific configuration space of the internal virtual PCI-to-PCI bridges with subordinate PCI Express hierarchies. The BIOS must configure each active root port to power-down. When the configuration write is received to set the “PM Turn Off” bit, the associated root port transmits a PM_Turn_Off message downstream. At this point, any traffic in-flight continues to be handled normally by the IMCH – routed outbound, and completed inbound.
3. The target PCI Express device ceases generation of new transactions inbound, waits for all pending transactions to complete and prepares to lose power and clocking. If the target device has a subordinate hierarchy of its own, it propagates the PM_Turn_Off message downstream and waits for acknowledges from all subordinate ports. Once ready to be brought off-line, the target device issues a PM_TO_Ack TLP cycle in acknowledgement back to the root. Note that the link is still communicative at this point, with both power and clock available.
4. After issuing the PM_TO_Ack cycle, the downstream device then issues a PM_Enter_L23 DLLP continuously upstream until it receives an acknowledge. In response to the PM_TO_Ack, the root port will commence the PCI Express handshake sequence necessary to sets its “Turn Off Ack” status bit. In response to the PM_Enter_L23 DLLP, the root transitions its downstream link to the electrical idle state. (This protocol sequence directly mirrors the L1 entry sequence.)
5. ACPI BIOS, which has been waiting for all of the “Turn Off Ack” status bits to assert, now clears all the command and status bits associated with the PME_TURN_OFF. The routine then informs the power manager to go ahead with the change to the system power state.
   Note that software is required (by the PCI Express specification) to implement a “dead-man” timer such that a failure to receive a full complement of “Turn Off Ack” status bits does not result in an indefinite hang. This timeout is nominally 1 second, after which the power state change proceeds regardless.
6. The power manager drops power and clocking to the target device(s), and all associated links automatically transition to either the L2 or L3 uncommunicative power states. The links enter L2 if Vaux is supplied by the platform, otherwise they enter L3. (Note that the IMCH does not support Vaux, so all downstream lanes will necessarily go to the L3 state.) The platform remains in the low-power state until a wake event is signaled.

In a fully PCI Express aware core logic implementation, the ACPI BIOS would not need to act as the interlock between the IMCH and the power manager, as all that functionality would be handled in hardware via direct messaging.

§ §

16.0 IMCH Registers

16.1 IMCH Registers: Bus 0, Device 0, Function 0

The Integrated Memory Controller Hub (IMCH) registers are in Bus 0, Device 0, Function 0. Table 16-1 provides the register address map for this device and function.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Table 16-1. Bus 0, Device 0, Function 0: Summary of IMCH PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID – Vendor Identification Register" on page 391	8086h
02h	03h	"Offset 02h: DID – Device Identification Register" on page 391	5020h
04h	05h	"Offset 04h: PCICMD: PCI Command Register" on page 392	0006h
06h	07h	"Offset 06h: PCISTS: PCI Status Register" on page 393	0010h
08h	08h	"Offset 8h: RID - Revision Identification Register" on page 394	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 394	00h
0Bh	0Bh	"Offset 0Bh: BCC – Base Class Code Register" on page 394	06h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 395	80h
14h	17h	"Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 396	00000000h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 396	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 397	0000h
4Ch	4Fh	"Offset 4Ch: NSIBAR - Root Complex Block Address Register" on page 397	00000000h
50h	50h	"Offset 50h: CFG0- IMCH Configuration 0 Register" on page 398	0Ch
51h	51h	"Offset 51h: IMCH_CFG1 – IMCH Configuration 1 Register" on page 399	00000h
53h	53h	"Offset 53h: CFGNS1 - Configuration 1 (Non-Sticky) Register" on page 399	00h
58h	58h	"Offset 58h: FDHC - Fixed DRAM Hole Control Register" on page 400	00h
59h	59h	"Offset 59h: PAM0 - Programmable Attribute Map 0 Register" on page 401	00h
5Ah	5Ah	"Offset 5Ah: PAM1: Programmable Attribute Map 1 Register" on page 402	00h
5Bh	5Bh	"Offset 5Bh: PAM2 - Programmable Attribute Map 2 Register" on page 403	00h
5Ch	5Ch	"Offset 5Ch: PAM3 - Programmable Attribute Map 3 Register" on page 404	00h
5Dh	5Dh	"Offset 5Dh: PAM4 - Programmable Attribute Map 4 Register" on page 405	00h
5Eh	5Eh	"Offset 5Eh: PAM5 - Programmable Attribute Map 5 Register" on page 406	00h
5Fh	5Fh	"Offset 5FH: PAM6 - Programmable Attribute Map 6 Register" on page 407	00h

Table 16-1. Bus 0, Device 0, Function 0: Summary of IMCH PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
9Ch	9Ch	"Offset 9Ch: DEVPRES - Device Present Register" on page 408	33h
9Dh	9Dh	"Offset 9Dh: EXSMRC - Extended System Management RAM Control Register" on page 409	00h
9Eh	9Eh	"Offset 9Eh: SMRAM - System Management RAM Control Register" on page 411	02h
9Fh	9Fh	"Offset 9Fh: EXSMRAMC - Expansion System Management RAM Control Register" on page 413	07h
B8h	BBh	"Offset B8h: IMCH_MENCBASE: IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Base Address Register" on page 413	000FFFFFh
BCh	BFh	"Offset BCh: IMCH_MENCLIMIT - IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Limit Address Register" on page 414	00000000h
C4h	C5h	"Offset C4h: TOLM - Top of Low Memory Register" on page 415	0800h
C6h	C7h	"Offset C6h: REMAPBASE - Remap Base Address Register" on page 416	03FFh
C8h	C9h	"Offset C8h: REMAPLIMIT – Remap Limit Address Register" on page 416	0000h
CAh	CBh	"Offset CAh: REMAPOFFSET - Remap Offset Register" on page 417	0000h
CCh	CDh	"Offset CCh: TOM - Top Of Memory Register" on page 417	0000h
CEh	CFh	"Offset CEh: HECBASE - PCI Express Port A (PEA) Enhanced Configuration Base Address Register" on page 418	E000h
D8h	D8h	"Offset D8h: CACHECTL0 - Write Cache Control 0 Register" on page 418	00h
DEh	DFh	"Offset DEh: SKPD - Scratchpad Data Register" on page 419	0000h
F6h	F6h	"Offset F6h: IMCH_TST2 - IMCH Test Byte 2 Register" on page 419	00h
60h at 1h	60h at 1h	"Offset 60h: DRB[0-3] - DRAM Row [3:0] Boundary Register" on page 421	ffh
70h at 4h	73h at 4h	"Offset 70h: DRA[0-1] - DRAM Row [0:1] Attribute Register" on page 422	00000515h
78h	7Bh	"Offset 78h: DRT0 - DRAM Timing Register 0" on page 424	242AD280h
64h	67h	"Offset 64h: DRT1 - DRAM timing Register 1" on page 431	12110000h
7Ch	7Fh	"Offset 7Ch: DRC - DRAM Controller Mode Register" on page 435	00000002h
84h	87h	"Offset 84h: ECCDIAG - ECC Detection/Correction Diagnostic Register" on page 437	00000000h
88h	8Bh	"Offset 88h: SDRC - DDR SDRAM Secondary Control Register" on page 439	00000002h
8Ch	8Ch	"Offset 8Ch: CKDIS - CK/CK# Clock Disable Register" on page 441	00h
8Dh	8Dh	"Offset 8Dh: CKEDIS - CKE Clock Enable Register" on page 442	00h
90h	93h	"Offset 90h: SPARECTL - SPARE Control Register" on page 443	00000000h
B0h	B3h	"Offset B0h: DDR2ODTC - DDR2 ODT Control Register" on page 444	00000000h

16.1.1 Register Details

16.1.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 16-2. Offset 00h: VID – Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:**	8086h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel 8086h.		8086h	RO

16.1.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Table 16-3. Offset 02h: DID – Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:**	5020h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IMCH Host-NSI Bridge Function 0.		5020h	RO

16.1.1.3 Offset 04h: PCICMD: PCI Command Register

Since IMCH Device 0 does not physically reside on a PCI bus, many of the bits are not supported.

Table 16-4. Offset 04h: PCICMD: PCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0006h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09	FB2B	**Fast Back-to-Back Enable:** This bit is hardwired to 0.		0b	RO
08	SERRE	**SERR Enable:** This bit is a global enable bit for Device 0 SERR messaging. The IMCH does not have a SERR signal. The IMCH communicates the SERR condition by sending a SERR message over NSI to the IICH. 0 = Disable. The SERR message is not generated by the IMCH for Device 0. 1 = Enable. The IMCH enables generation of SERR messages over NSI for specific Device 0, Function 0 error conditions that are enabled via the PCICMD register. The error status is reported in the PCISTS registers. The only error event enabled through Device 0, Function 0 is Detected Parity Error which is essentially a NSI poisoned TLP, and is enabled by the parity error enable bit (PERRE). ***Note:*** This bit only controls SERR messaging for Device 0, Function 0. Device 0, Function 1, and Devices 1-7 have their own SERR bits to control error reporting for error conditions occurring on their respective devices. The control bits are used in a logical OR manner to enable the SERR NSI message mechanism.		0b	RW
07	Reserved	Reserved		0b	
06	PERRE	**Parity Error Enable:** 0 = Disable. The IMCH does not take any action when it detects data corruption on NSI. 1 = Enable. The IMCH generates an SERR message over the NSI to the IICH when a poisoned TLP is detected by the IMCH on NSI (DPE set in PCISTS) and SERRE is set to 1.		0b	RW
05 : 03	Reserved	Reserved		0h	
02	BME	**Bus Master Enable:** The IMCH is always enabled as a master on NSI. This bit is hardwired to 1. Writes to this bit position have no effect.		1b	RO
01	MAE	**Memory Access Enable:** This bit is hardwired to 1.		1b	RO
00	Reserved	Reserved		0b	

16.1.1.4 Offset 06h: PCISTS: PCI Status Register

PCISTS is a 16-bit status register that reports the occurrence of error events on Device 0's PCI interface. Since IMCH Device 0 does not physically reside on a PCI bus many of the bits are not supported.

Table 16-5. Offset 06h: PCISTS: PCI Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0010h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** This bit is set to 1 whenever it receives a poisoned TLP regardless of the state of the parity error response bit. Software may clear this by writing a 1 to this bit.		0b	RWC
14	SSE	**Signaled System Error:** 0 = Software clears this bit by writing a 1 to the bit location. 1 = IMCH Device 0, Function 0 generates a SERR message over NSI for any enabled Device 0, Function 0 error condition. Device 0 error conditions are enabled in the PCICMD register. Device 0 error flags are read/reset from the PCISTS register. The only error that can be enabled to signal system error through Device 0, Function 0 is the detected parity error which is essentially a NSI poisoned TLP. Software may clear this by writing a 1 to this bit.		0b	RWC
13	RMAS	Received Master Abort Status: This bit is set if the IMCH generates a NSI request that receives a completion with unsupported request completion status. Software may clear this by writing a 1 to this bit.		0b	RWC
12	RTAS	**Received Target Abort Status:** Set to 1 by hardware if the IMCH generated a request that received a completion with Completer Abort status. Software clears this bit by writing a 1 to this bit location.		0b	RWC
11	STAS	**Signaled Target Abort Status:** The IMCH does not generate a Completer Abort on the NSI completion packet. This bit is hardwired to w10. Writes to this bit position have no effect.		0b	RO
10 : 09	Reserved	Reserved		00b	
08	DPD	**Master Data Parity Error Detected:** This bit is hardwired to 0.		0b	RWC
07	FB2B	**Fast Back-to-Back:** Reserved.		0b	
06 : 05	Reserved	Reserved		00b	
04	CLIST	**Capability List:** This bit is hardwired to 1 to indicate to the configuration software that this device/function implements a list of new capabilities. A list of new capabilities is accessed via register CAPPTR at configuration address offset 34h.		1b	RO
03 : 00	Reserved	Reserved		0h	

16.1.1.5 Offset 08h: RID - Revision Identification Register

This register contains the revision number of the IMCH Device 0. These bits are read-only and writes to this register have no effect.

Table 16-6. Offset 8h: RID - Revision Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:**	Variable		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the IMCH Device 0.		Variable	RO

16.1.1.6 Offset 0Ah: SUBC - Sub-Class Code Register

Table 16-7. Offset 0Ah: SUBC - Sub-Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SUBC	**Sub-Class Code:** This value indicates the Sub Class Code into which the IMCH Device 0 falls. 00h = Host Bridge		00h	RO

16.1.1.7 Offset 0Bh: BCC – Base Class Code Register

Table 16-8. Offset 0Bh: BCC – Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:**	06h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BASEC	**Base Class Code:** This value indicates the Base Class Code for the IMCH Device 0. 06h = Bridge device		06h	RO

16.1.1.8 Offset 0Eh: HDR - Header Type Register

Table 16-9. Offset 0Eh: HDR - Header Type Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:**	80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	***Note:*** **PCI Header:** The header type of the IMCH Device 0. 80h = multi-function device with standard header layout. This register should return a 00h indicating a single function device, when both functions 1 and 2 are disabled.			80h	RO

16.1.1.9 Offset 14h: SMRBASE - System Memory RCOMP Base Address Register

The SMRBASE is a standard PCI Base Address register that is used to set the base of the Memory Mapped Registers used to control the System Memory I/O Buffer RCOMP. In addition to calibration, the DCAL engine which resides within this memory mapped region also performs RAS functions. In addition, there are some BIOS scratch registers within this region. The actual behavior of this register depends on the SMRCOMP MMR Enable bit in the IMCH TST2 register (Device 0, Function 0, Offset F6, Bit 6). When IMCH TST2[6] is set, this register behaves like a standard PCI BAR requesting 4 Kbyte of address space. When IMCH TST2[6] is clear, this register is hardwired to all zeros, effectively disabling the BAR and the corresponding SM MMR region. Because of the more extensive functionality supported by DCAL, it is expected that once this address space has been enabled by System BIOS, it remains enabled to support various RAS features.

Note: All accesses to these Memory Mapped Registers must be made as a single Dword (4 bytes) or less. Access must be aligned on a natural boundary.

Table 16-10. Offset 14h: SMRBASE - System Memory RCOMP Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	UPBITS	**Upper Programmable Base Address:** These bits are part of the SM MMR region, normally set by configuration software to locate the base address of the region. The actual behavior of this field depends on the SM MMR Enable bit in the IMCH TST2 register (bit 6) as defined above. When IMCH TST2[6] = 1 these bits are Read/Write. When IMCH TST2[6] = 0 these bits are Read-Only as zeros.		00000h	RW or RO
11 : 04	LOWBITS	**Lower Bits:** These bits are hardwired to 0. This forces the size of the memory region to be 4 Kbyte.		00h	RO
03	PF	**Prefetchable:** This bit is hardwired to 0 to indicate that the System Memory MMR region is NON-Prefetchable.		0b	RO
02 : 01	TYPE	**Addressing Type:** These bits determine addressing type and they are hardwired to 00 to indicate that address range defined by the upper bits of this register can be located anywhere in the 32-bit address space in order to comply with the PCI specification for base address registers.		00b	RO
00	MSPACE	**Memory Space Indicator:** Hardwired to 0 to identify the MMR range as a memory range as per the specification for PCI base address registers.		0b	RO

16.1.1.10 Offset 2Ch: SVID - Subsystem Vendor Identification Register

This value is used to identify the vendor of the subsystem.

Table 16-11. Offset 2Ch: SVID - Subsystem Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBVID	**Subsystem Vendor ID:** This field must be programmed during boot-up to indicate the vendor of the system board.		0000h	RWO

16.1.1.11 Offset 2Eh: SID - Subsystem Identification Register

This value is used to identify a particular subsystem.

Table 16-12. Offset 2Eh: SID - Subsystem Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBID	**Subsystem ID:** This field must be programmed during BIOS initialization. After it has been written once it becomes Read-Only. When any byte or combination of bytes of this register is written, the register value locks and cannot be further updated.		0000h	RWO

16.1.1.12 Offset 4Ch: NSIBAR - Root Complex Block Address Register

This is the base address for the Root Complex memory-mapped configuration space. This window of addresses contains the Root Complex Register Block for the NSI hierarchy associated with the IMCH. There is no physical memory within this 4 Kbyte window that can be addressed. The 4 Kbyte reserved by this register does not alias to any PCI 2.3 compliant memory mapped space.

All accesses to these Memory Mapped Registers must be made as a single Dword (4 bytes) or less. Access must be aligned on a natural boundary.

Table 16-14. Offset 4Ch: NSIBAR - Root Complex Block Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 4Ch **Offset End:** 4Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	NSI_BA	NSI **Base Address:** The BIOS programs this register resulting in a base address for a 4 Kbyte block of contiguous memory address space. This register ensures that a naturally aligned 4 Kbyte space is allocated within total addressable memory space of 4 Gbyte. System Software uses this base address to program the NSI register set. When IMCH TST2[5] = 1, the NSI Memory Mapped Register space is visible and memory mapped accesses are claimed and decoded appropriately. When IMCH TST2[5] = 0, the NSI Memory Mapped Register space is disabled and does not claim any memory. (THE NSIBAR register is still read/write accessible.)		00000h	RW or RO
11 : 00	Reserved	Hardwired to 0		000h	

16.1.1.13 Offset 50h: CFG0 - IMCH Configuration 0 Register

MCHCFG consists of IMCH CFG1 in the upper 8 bits and IMCH CFG0 in the lower 8 bits.

Table 16-15. Offset 50h: CFG0- IMCH Configuration 0 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 50h **Offset End:** 50h	
Size: 8 bit	**Default:**	0Ch		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		0h	
02	IOQD	**In-Order Queue Depth:** This bit reflects the value sampled on HA[7]# on the de-assertion of the CPURST#. It indicates the depth of the CPU bus in-order queue. 0 = HA[7]# has been sampled asserted (e.g., logic one, or electrical low). The depth of the IOQ is set to one (e.g., no pipelining on the processor bus). HA[7]# may be driven low during CPURST# by an external source. 1 = HA[7]# was sampled as deasserted (e.g. logic zero or electrical high). The depth of the processor bus in-order queue is configured to the maximum (e.g., 12).		1b	RO
01	DRFD	**Deferred Resource Fairness Disable:** 0 = Clearing the bit allows the fairness logic to start working again. 1 = Setting this bit clears the fairness logic for deferred resources and hold it in reset. ***Note:*** This bit should only be changed in the event that there is some issue with the fairness logic.		0b	RW
00	Reserved	Reserved		0b	

16.1.1.14 Offset 51h: IMCH_CFG1 – IMCH Configuration 1 Register

Table 16-16. Offset 51h: IMCH_CFG1 – IMCH Configuration 1 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 51h **Offset End:** 51h	
Size: 8 bit	**Default:**	00000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	NSG	**Number of Stop Grant Cycles:** Number of Stop Grant transactions expected on the FSB bus before a Req_C2 packet is sent to the IICH. This field is programmed by the BIOS after it has enumerated the processors and before it has enabled Stop Clock generation in the IICH. Once this field has been set, it must not be modified. Note that each enabled thread within each CPU generates Stop Grant Acknowledge transactions. ***Note:*** This register is read/write and not write-once as in some implementations. Encoding Description 0 0 0 NSI Stop Grant generated after 1 FSB Stop Grant 0 0 1 NSI Stop Grant generated after 2 FSB Stop Grant 0 1 0 NSI Stop Grant generated after 3 FSB Stop Grant 0 1 1 NSI Stop Grant generated after 4 FSB Stop Grant 1 0 0 NSI Stop Grant generated after 5 FSB Stop Grant 1 0 1 NSI Stop Grant generated after 6 FSB Stop Grant 1 1 0 NSI Stop Grant generated after 7 FSB Stop Grant 1 1 1 NSI Stop Grant generated after 8 FSB Stop Grant		000b	RW
04 : 00	Reserved	Reserved		00000b	

16.1.1.15 Offset 53h: CFGNS1 - Configuration 1 Register

This register contains IMCH control bits that are not sticky.

Table 16-17. Offset 53h: CFGNS1 - Configuration 1 (Non-Sticky) Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 53h **Offset End:** 53h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		000b	
01	THWO	**Throttled-Write Occurred:** 0 = Writing a zero clears this bit. 1 = This bit is set by hardware when a write is throttled. This happens when the maximum allowed number of writes has been reached during a time-slice and there is at least one more write to be completed.		0b	RW0C

Table 16-17. Offset 53h: CFGNS1 - Configuration 1 (Non-Sticky) Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** 53h **Offset End:** 53h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
00	THRO	**Throttled-Read Occurred:** 0 = Writing a zero clears this bit. 1 = This bit is set by hardware when a read is throttled. This happens when the maximum allowed number of reads has been reached during a time-slice and there is at least one more read to be done.			0b	RW0C

16.1.1.16 Offset 58h: FDHC - Fixed DRAM Hole Control Register

This 8-bit register controls a fixed DRAM hole from 15–16 Mbytes.

Table 16-18. Offset 58h: FDHC - Fixed DRAM Hole Control Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** 58h **Offset End:** 58h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07	HEN	**Hole Enable:** This field enables a memory hole in DRAM space. The DRAM that lies “behind” this space is not remapped. 0 = No memory hole 1 = Memory hole from 15–16 Mbytes. Accesses in this range are sent to NSI.			0b	RW
06 : 00	Reserved	Reserved			00h	

16.1.1.17 Offset 59h: PAM0 - Programmable Attribute Map 0 Register

This register controls the read, write, and shadowing attributes of the BIOS area from 0F0000h-0FFFFFh. See Section 19.0.3, "PAM Memory Spaces" for more information on PAM memory spaces.

Table 16-19. Offset 59h: PAM0 - Programmable Attribute Map 0 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 59h **Offset End:** 59h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register:** This field controls the steering of read and write cycles that address the BIOS area from 0F0000 to 0FFFFF. Encoding Description: 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write-Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 00	Reserved	Reserved		0h	

16.1.1.18 Offset 5Ah: PAM1 - Programmable Attribute Map 1 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0C0000h-0C7FFFh.

Table 16-20. Offset 5Ah: PAM1: Programmable Attribute Map 1 Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 5Ah **Offset End:** 5Ah	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0C4000-0C7FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0C4000 to 0C7FFF Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write-Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0C0000-0C3FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0C0000 to 0C3FFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write-Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW

16.1.1.19 Offset 5Bh: PAM2 - Programmable Attribute Map 2 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0C8000h-0CFFFFh.

Table 16-21. Offset 5Bh: PAM2 - Programmable Attribute Map 2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 5Bh **Offset End:** 5Bh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0CC000-0CFFFF:** Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write-Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0C8000-0CBFFF:** This field controls the steering of read and write cycles that address the BIOS area from 0C8000 to 0CBFFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. A writes are forwarded to NSI. 1 0 Write-Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW

16.1.1.20 Offset 5Ch: PAM3 - Programmable Attribute Map 3 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0D0000h-0D7FFFh.

Table 16-22. Offset 5Ch: PAM3 - Programmable Attribute Map 3 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 5Ch **Offset End:** 5Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0D4000-0D7FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0D4000 to 0D7FFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0D0000-0D3FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0D0000 to 0D3FFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW

16.1.1.21 Offset 5Dh: PAM4 - Programmable Attribute Map 4 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0D8000h-0DFFFFh.

Table 16-23. Offset 5Dh: PAM4 - Programmable Attribute Map 4 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 5Dh **Offset End:** 5Dh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0DC000-0DFFFF:** This field controls the steering of read and write cycles that address the BIOS area from 0DC000 to 0DFFFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0D8000-0DBFFF:** This field controls the steering of read and write cycles that address the BIOS area from 0D8000 to 0DBFFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW

16.1.1.22 Offset 5Eh: PAM5 - Programmable Attribute Map 5 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0E0000h-0E7FFFh.

Table 16-24. Offset 5Eh: PAM5 - Programmable Attribute Map 5 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 5Eh **Offset End:** 5Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0E4000-0E7FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0E4000 to 0E7FFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0E0000-0E3FFF:** This field controls the steering of read and write cycles that address the BIOS area from 0E0000 to 0E3FFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI. 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI. 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI. 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM.		00b	RW

16.1.1.23 Offset 5Fh: PAM6 - Programmable Attribute Map 6 Register

This register controls the read, write, and shadowing attributes of the BIOS areas from 0E8000h-0EFFFFh.

Table 16-25. Offset 5FH: PAM6 - Programmable Attribute Map 6 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 5Fh **Offset End:** 5Fh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00b	
05 : 04	HIENABLE	**Attribute Register 0EC000-0EFFFF:** This field controls the steering of read and write cycles that address the BIOS area from 0EC000 to 0EFFFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM		00b	RW
03 : 02	Reserved	Reserved		00b	
01 : 00	LOENABLE	**Attribute Register 0E8000-0EBFFF:** This field controls the steering of read and write cycles that address the BIOS area from 0E8000 to 0EBFFF. Encoding Description 0 0 DRAM Disabled - All accesses are directed to NSI 0 1 Read-Only - All reads are serviced by DRAM. All writes are forwarded to NSI 1 0 Write Only - All writes are sent to DRAM. Reads are serviced by NSI 1 1 Normal DRAM Operation - All reads and writes are serviced by DRAM		00b	RW

16.1.1.24 Offset 9Ch: DEVPRES - Device Present Register

The Device Present bits can be used to enable/disable devices within the IMCH and make their PCI configuration space respectively visible/invisible to software. The Device Present bits convey when cleared that the corresponding device is never available.

When a bit is 0, the configuration space associated with that device is hidden, returning all 1's for all configuration register reads just as if the cycle terminated with a master abort on PCI. For the two PCIe* devices listed the I/O buffers and compensation associated with those devices are disabled and tri-stated. When a bit is 1, the configuration space associated with that device is accessible. For the two PCIe* devices the I/O buffers and compensation are enabled.

Note: BIOS should write this register as part of its power on configuration sequence.

Bits within this register are broken into two categories "RWO or RO" and "RW" and are functionally defined below.

"RWO or RO" bits:

All of the "RWO or RO" bits in this register are gated with a SKU value. The SKU value has priority over values written by software. This means that if the SKU value of this bit is set to a 0, neither a configuration write or a reset sets this bit.

If a SKU value is not present then the register will have RO access.

When a SKU value is not present, the device is invisible to software. Writes to this register when the SKU value is not present, will have no effect, always returning '0' when read.

If a SKU value is present the register will have RWO access.

Note: BIOS must write a bit to a 0 to disable a device, or a 1 to enable a device. This register can only be written once. After the first write, the register is locked.

Note: RWO Devices that are disabled via software can only be re-enabled via a reset. This register should only be written to at boot time when there is no traffic to or from the PEA.

Once software or BIOS has written these RWO register bits for the first time after power-up, the register value locks, and cannot be further updated.

In other words, once software has disabled RWO devices, they can only be re-enabled via a reset. For RWO access bits the IMCH does not support turning off a device, and then turning it back on. (The reverse is also true: once software has enabled RWO devices, they can only be re-disabled via a reset.)

RW bits:

Two devices have RW bit access. These devices need to be enabled/disabled for power management during normal operation

Table 16-26. Offset 9Ch: DEVPRES - Device Present Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 9Ch **Offset End:** 9Ch	
Size: 8 bit	**Default:** 33h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		0b	
05	Reserved	Reserved		1b	RW
04	Device_4_Present	0 = PCI-to-PCI Bridge is disabled. 1 = PCI-to-PCI Bridge is enabled.		1b	RW
03	Device_3_Present	0 = PCI Express* port A1 (x4) is disabled. In this state, port A (Device 2) can operate with a maximum x8 link width. 1 = PCI Express port A1 is enabled. In this state, port A can operate with a maximum x4 link width.When the SKU value is cleared, this field is read/write. When the SKU value is set, this field becomes a read-only '0'		0b	RWO or RO

Table 16-26. Offset 9Ch: DEVPRES - Device Present Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 9Ch **Offset End:** 9Ch	
Size: 8 bit	**Default:**	33h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	Device_2_Present	0 = PCI Express port A is disabled. 1 = PCI Express port A is enabled.When the SKU value is cleared, this field is read/write. When the SKU value is set, this field becomes a read-only '0'		0b	RWO or RO
01	Device_1_Present	0 = EDMA Controller is disabled. 1 = EDMA Controller is enabled.		1b	RWO
00	Reserved	Reserved		1b	

16.1.1.25 Offset 9Dh: EXSMRC - Extended System Management RAM Control Register

The Extended SMRAM register controls the configuration of Extended SMRAM space. The Extended SMRAM (E_SMRAM) memory provides a write-back cacheable SMRAM memory space that is above 1 MByte.

Table 16-27. Offset 9Dh: EXSMRC - Extended System Management RAM Control Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 9Dh **Offset End:** 9Dh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	H_SMRAME	**Enable High SMRAM:** Controls the SMM memory space location (above 1 MByte or below 1 MByte) 0 = High SMRAM memory space is disabled. 1 = And G_SMRAME is 1, the high SMRAM memory space is enabled. SMRAM accesses within the range 0FEDA_0000h to 0FEDB_FFFFh are remapped to DRAM addresses within the range 000A0000h to 000BFFFFh. Once D_LCK (See Table 35) has been set, this bit becomes Read-Only.		0b	RWL

Table 16-27. Offset 9Dh: EXSMRC - Extended System Management RAM Control Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 9Dh **Offset End:** 9Dh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	MDAP	**MDA Present:** This bit works with the VGA Enable bits in the BCTRL registers of Devices 2–3 to control the routing of CPU initiated transactions targeting MDA compatible I/O and memory address ranges. This bit should not be set if none of the VGA Enable bits are set. If none of the VGA enable bits are set, then accesses to IO address range x3BCh-x3BFh are forwarded to NSI. If the VGA enable bit is not set then accesses to IO address range x3BCh-x3BFh are treated just like any other IO accesses. For example, the cycles are forwarded to PEA[0:1] if the address is within the corresponding IOBASE and IOLIMIT and ISA enable bit is not set, otherwise they are forwarded to NSI. ***Note:*** Since the logic performs the address decoding on a DW boundary, the DW that includes the address 3BF also includes addresses 3BC, 3BD, and 3BE, and accesses to any of these byte addresses are handled as MDA references. MDA resources are defined as the following: Memory: 0B0000h - 0B7FFFh I/O: 3B4h, 3B5h, 3B8h, 3B9h, 3BAh, 3BFh, (Including ISA address aliases, A[15:10] are not used in decode) ***Note:*** The VGA region includes I/O space ranges 3B0-3BBh, and 3C0-3DFh, so there is an overlap between these two I/O regions. Any I/O reference that includes the I/O locations listed above, or their aliases, are forwarded to NSI even if the reference includes I/O locations not listed above. The following table shows the behavior for all combinations of MDA and VGA: **VGA MDA Behavior** 0 0 All References to MDA and VGA go to NSI 0 1 Illegal Combination (DO NOT USE) 1 0 All References to VGA go to device with VGA enable set. MDA- only references (I/O address 3BF and aliases) go to NSI. 1 1 VGA-only references go to the PCI Express port which has its VGA Enable bit set. MDA references go to the NSI.		0b	RW
05	APICDIS	**APIC Memory Range Disable:** 0 = The IMCH send cycles between 0_FEC0_0000 and 0_FEC7_FFFF to NSI, accesses between 0_FEC8_0000 and 0_FEC8_0FFF are sent to PEA0, between 0_FEC8_1000 and 0_FEC8_1FFF are sent to PEA1B. 1 = The IMCH forwards all accesses to the IOAPIC regions to NSI.		0b	RW
04	Reserved	Reserved		0b	
03	G_SMRAME	**Global SMRAM Enable:** 0 = The Compatible SMRAM functions are disabled. 1 = The Compatible SMRAM functions are enabled, providing 128 Kbyte of DRAM accessible at the A0000h address while in SMM (ADS# with SMM decode). To enable Extended SMRAM function this bit has be set to 1. Refer to Section 16.1.1.26, "Offset 9Eh: SMRAM - System Management RAM Control Register" for more details. Once D_LCK (See Table 16-28) is set, this bit becomes read-only.		0b	RWL

Table 16-27. Offset 9Dh: EXSMRC - Extended System Management RAM Control Register (Sheet 3 of 3)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** 9Dh **Offset End:** 9Dh	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
02 : 01	TSEG_SZ	**TSEG Size:** Selects the size of the TSEG memory block if enabled. Memory from the top of DRAM space (TOLM - TSEG_SZ) to TOLM is partitioned away so that it may only be accessed by the processor interface and only then when the SMM bit is set in the request packet. Non-SMM accesses to this memory region are specially terminated when the TSEG memory block is enabled. Note that once D_LCK (See Table 16-28) is set, these bits become Read-Only. 0 0 (TOLM – 128 k) to TOLM 0 1 (TOLM – 256 k) to TOLM 1 0 (TOLM – 512 k) to TOLM 1 1 (TOLM – 1 M) to TOLM			00b	RWL
00	T_EN	**TSEG Enable:** Enabling of SMRAM memory for Extended SMRAM space only. 0 = SMRAM memory for Extended SMRAM space disabled. 1 = And G_SMRAME =1 and T_EN = 1, the TSEG is enabled to appear in the appropriate physical address space. Once D_LCK (See Table 16-28) is set, this bit becomes Read-Only.			0b	RWL

16.1.1.26 Offset 9Eh: SMRAM - System Management RAM Control Register

The SMRAMC register controls how accesses to Compatible and Extended SMRAM spaces are treated. The Open, Close, and Lock bits function only when G_SMRAME bit is set to a 1. Also, the OPEN bit must be reset before the LOCK bit is set.

Table 16-28. Offset 9Eh: SMRAM - System Management RAM Control Register (Sheet 1 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** 9Eh **Offset End:** 9Eh	
Size: 8 bit	**Default:**	02h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved			0b	
06	D_OPEN	**SMM Space Open:** 0 = The SMM space DRAM is not visible 1 = And D_LCK=0, the SMM space DRAM is made visible even when SMM decode is not active. This is intended to help BIOS initialize SMM space. Software must ensure that D_OPEN=1 and D_CLS=1 are not set at the same time. This bit becomes RO when D_LCK is set to 1.			0b	RWL

Table 16-28. Offset 9Eh: SMRAM - System Management RAM Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 9Eh **Offset End:** 9Eh	
Size: 8 bit	**Default:**	02h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	D_CLS	**SMM Space Closed:** 0 = SMM space DRAM is accessible to data references 1 = SMM space DRAM is not accessible to data references, even if SMM decode is active. Code references may still access SMM space DRAM. This allows SMM software to reference through SMM space to update the display even when SMM is mapped over the VGA range. Software must ensure that D_OPEN=1 and D_CLS=1 are not set at the same time. ***Note:*** The D_CLS bit only applies to Compatible SMM space.		0b	RW
04	D_LCK	**SMM Space Locked:** 0 = SMM space unlocked 1 = And then D_OPEN is reset to 0 and D_LCK, D_OPEN, H_SMRAME, TSEG_SZ and T_EN become Read-Only. D_LCK can be set to 1 via a normal configuration space write but can only be cleared by a Full Reset. The combination of D_LCK and D_OPEN provide convenience with security. The BIOS can use the D_OPEN function to initialize SMM space and then use D_LCK to lock SMM space in the future so that no application software (or BIOS itself) can violate the integrity of SMM space, even if the program has knowledge of the D_OPEN function.		0b	RWS
03	Reserved	Reserved.		0b	
02 : 00	C_BASE_SEG	**Compatible SMM Space Base Segment:** This field indicates the location of SMM space. SMM DRAM is not remapped. It is simply made visible if the conditions are right to access SMM space, otherwise the access is treated as a VGA access. Since the IMCH supports only the SMM space between A0000 and BFFFF, this field is hardwired to 010.		010b	RO

16.1.1.27 Offset 9Fh: EXSMRAMC - Expansion System Management RAM Control Register

The Extended SMRAM register controls the configuration of Extended SMRAM space. The Extended SMRAM (E_SMRAM) memory provides a write-back cacheable SMRAM memory space that is above 1 MByte.

Table 16-29. Offset 9Fh: EXSMRAMC - Expansion System Management RAM Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 9Fh **Offset End:** 9Fh	
Size: 8 bit	**Default:**	07h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	E_SMERR	**Invalid SMRAM Access:** 0 = CPU has not accessed the defined memory ranges in Extended SMRAM. 1 = This bit is set when CPU has accessed the defined memory ranges in Extended SMRAM (High Memory and T-segment) while not in SMM space and with the D-OPEN bit = 0. It is software's responsibility to clear this bit. This bit is cleared by software writing a 1 to the bit location.		0b	RWC
06 : 03	Reserved	Reserved		0h	
02	SM_CACHE	**SMRAM Cacheable: This bit is forced to 1 by** IMCH**. (Moved from ESMRAMC bit 5)**		1b	RO
01	SM_L1	**L1 Cache Enable for SMRAM:** This bit is forced to 1 by IMCH. (Moved from ESMRAMC bit 4)		1b	RO
00	SM_L2	**L2 Cache Enable for SMRAM:** This bit is forced to 1 by IMCH. (Moved from ESMRAMC bit 3)		1b	RO

16.1.1.28 Offset B8h: IMCH_MENCBASE - IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Base Address Register

Table 16-30. Offset B8h: IMCH_MENCBASE: IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** B8h **Offset End:** BBh	
Size: 32 bit	**Default:**	000FFFFFh		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		000h	
19 : 00	MENCBASE	**IA/ASU Shared Non-Coherent Memory Base Address Bits[31:12]:** Specifies the address of the lower boundary of the IA/ASU shared non-coherent window in 32-bit system address space. The window is 4KB-aligned and inclusive of this address. This register field specifies bits[31:12] of the address; bits[11:0] are assumed zeros given 4KB alignment.		FFFFFh	RW

16.1.1.29 Offset BCh: IMCH_MENCLIMIT - IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Limit Address Register

Table 16-31. Offset BCh: IMCH_MENCLIMIT - IA/ASU Shared Non-Coherent (AIOC-Direct) Memory Limit Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** BCh **Offset End:** BFh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		00h	
19 : 00	MENCLIMIT	**IA/ASU Shared Non-Coherent Memory Limit Address Bits[31:12]:** Specifies the address of the upper boundary of the IA/ASU shared non-coherent window in 32-bit system address space. The window is 4KB-aligned and inclusive of this address. This register field specifies bits[31:12] of the address; bits[11:0] are assumed ones. Setting IMCH_MENCLIMIT less than IMCH_MENCBASE indicates a zero-sized window and thus that all memory is coherent.		00000h	RW

16.1.1.30 Offset C4h: TOLM - Top of Low Memory Register

This register contains the maximum address below 4 Gbyte that must be treated as a memory access and is defined on a 128 Mbyte boundary. Usually it is below the areas configured for PCI Express, NSI, and PCI memory. The memory address found in DRB3 reflects the amount of total memory populated.

Table 16-32. Offset C4h: TOLM - Top of Low Memory Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** C4h **Offset End:** C5h	
Size: 16 bit	**Default:** 0800h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	TOLM	**Top of Low Memory:** This register corresponds to bits 31 to 27 of the system address which is 1 greater than the maximum DRAM location below 4 Gbyte. Configuration software must set this value to either the maximum amount of memory in the system or to the minimum address allocated for PCI memory or the graphics aperture, whichever is smaller. Address bits 26:00 are assumed to be 0 for the purposes of address comparison. Addresses equal to or greater than the TOLM, and less than 4 G, are treated as non-memory accesses. All accesses less than the TOLM are treated as DRAM accesses (except for the 15–16 Mbyte or PAM gaps). This register must be set to at least 0800h, for a minimum of 128 Mbyte of DRAM. There is also a minimum of 128 Mbyte of PCI space, since this register is on a 128 Mbyte boundary. Configuration software must set this value to either the maximum amount of memory in the system (same as DRB3), or to the lower 128 Mbyte boundary of the Memory Mapped IO range, whichever is smaller. Programming example: 1100_0b = 3 Gbyte (assuming that DBR7 is set > 4 Gbyte): An access to 0_C000_0000h or above (but <4 Gbyte) is considered above the TOLM and therefore not to DRAM. It may go to one of the PEA ports or NSI or be subtracted and decoded to NSI. An access to 0_BFFF_FFFFh and below is considered below the TOLM and go to DRAM.		00001b	RW
10 : 00	Reserved	Reserved		000h	

16.1.1.31 Offset C6h: REMAPBASE - Remap Base Address Register

Note: This register should not be enabled since the IA-32 core only supports 32-bit addressing.

Table 16-33. Offset C6h: REMAPBASE - Remap Base Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** C6h **Offset End:** C7h	
Size: 16 bit	**Default:** 03FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 : 00	REMAPBASE	**Remap Base Address Bits [35:26]:** The value in this register defines the lower boundary of the Remap window. The Remap window is inclusive of this address. In the decoder A[25:0] of the Remap Base Address are assumed to be 0s. Thus the bottom of the defined memory range is aligned to a 64 Mbyte boundary. When the value in this register is greater than the value programmed into the Remap Limit register, the Remap window is disabled. This field defaults to 3FF.		3FFh	RW

16.1.1.32 Offset C8h: REMAPLIMIT – Remap Limit Address Register

Note: This register should not be enabled since the IA-32 core only supports 32-bit addressing.

Table 16-34. Offset C8h: REMAPLIMIT – Remap Limit Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** C8h **Offset End:** C9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 : 00	REMAPLIMIT	**Remap Limit Address Bits [35:26]:** The value in this register defines the upper boundary of the Remap window. The Remap window is inclusive of this address. In the decoder A[25:00] of the Remap Limit Address are assumed to be Fs. Thus the top of the defined range is one less than a 64 Mbyte boundary. When the value in this register is less than the value programmed into the Remap Base register, the Remap window is disabled.		00h	RW

16.1.1.33 Offset CAh: REMAPOFFSET - Remap Offset Register

This register contains the difference between the REMAPBASE and TOLM.Note: This register should not be enabled since the IA-32 core only supports 32-bit addressing.

Table 16-35. Offset CAh: REMAPOFFSET - Remap Offset Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** CAh **Offset End:** CBh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 : 00	REMAPOFFST	**Remap Offset:** This register contains the difference between the REMAPBASE and TOLM. This register value corresponds to address bits 35:26. It is used to translate the physical FSB address to the system memory address for accesses to the remap region.		000h	RW

16.1.1.34 Offset CCh: TOM - Top Of Memory Register

This register contains the effective size of memory. The value in this register hides any DIMMs that can't be directly addressed. BIOS determines the memory size reported to the OS using this register.

Table 16-36. Offset CCh: TOM - Top Of Memory Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** CCh **Offset End:** CDh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 9	Reserved	Reserved		00h	
08 : 00	TOM	**Top of Memory:** This register reflects the effective size of memory. These bits correspond to address bits 35:27. (128 Mbyte granularity) Bits 26:00 are assumed to be 0.		000h	RW

16.1.1.35 Offset CEh: HECBASE - PCI Express Port A (PEA) Enhanced Configuration Base Address Register

Configuration software reads this register to determine where the 256 Mbyte range of addresses resides for this particular host bridge. This register contains the base address of enhanced configuration memory.

Table 16-37. Offset CEh: HECBASE - PCI Express Port A (PEA) Enhanced Configuration Base Address Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** CEh **Offset End:** CFh	
Size: 16 bit	**Default:**	E000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 12	HECBASE	**PEA Enhanced Configuration Base:** This register contains the address that corresponds to bits 31 to 28 of the base address for PEA enhanced configuration space below 4 Gbyte. Configuration software reads this register to determine where the 256 Mbyte range of addresses resides for this particular host bridge. BIOS needs to write this register at boot time. Settings 0 and F are not valid. When any byte or combination of bytes of this register is written, the register value locks down and cannot be further updated.			1110b	RWO
11 : 00	Reserved	Reserved.			000h	

16.1.1.36 Offset D8h: CACHECTL0 - Write Cache Control 0 Register

Table 16-38. Offset D8h: CACHECTL0 - Write Cache Control 0 Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0		**Offset Start:** D8h **Offset End:** D8h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved	Reserved for other customer visible features.			00h	RO
00	WCFLUSH	**Write Cache Flush:** 0 = Cleared by hardware when flush is complete. 1 = All entries in the write cache are flushed to DRAM with high priority. The arbiter no longer accepts requests until the write cache has been flushed. Software can poll this bit to determine when the flush is complete.			0b	RWS

16.1.1.37 Offset DEh: SKPD - Scratchpad Data Register

Table 16-39. Offset DEh: SKPD - Scratchpad Data Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SCRTCH	**Scratchpad:** These bits are simply Read/Write storage bits that have no effect on the IMCH functionality. BIOS typically programs this register to the revision ID of the Memory Reference Code.		0000h	RW

16.1.1.38 Offset F6h: IMCH_TST2 - IMCH Test Byte 2 Register

Table 16-40. Offset F6h: IMCH_TST2 - IMCH Test Byte 2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** F6h **Offset End:** F6h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0b	
06	SYSMMREN	**System Memory MMR Enable:** 0 = This BAR is hardwired to all zeros (effectively disabling this memory space). 1 = The SM Memory Mapped Register space and corresponding Base Address Register (B:D:F:R 0,0,0,14H) is visible. Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register"		0b	RW
05	NSIMMREN	NSI **MMR Enable:** 0 = The NSI Memory Mapped Register space is disabled, and does not claim any memory. (THE NSIBAR register is still read/write accessible.) 1 = The NSI Memory Mapped Register space is visible, and memory mapped accesses are claimed and decoded appropriately. (B:D:F:R 0,0,0,4CH) Section 16.1.1.12, "Offset 4Ch: NSIBAR - Root Complex Block Address Register"		0b	RW
04	Reserved	Reserved.		0b	
03	Reserved	Reserved		0b	
02	Reserved	Reserved		0b	
01	Reserved	Reserved		0b	
00	Reserved	Reserved		0b	

16.1.1.39 Offset 60h: DRB[0-3] – DRAM Row [3:0] Boundary Register

DRAM row boundary register defines the upper boundary address for each DRAM row with a granularity of 64MB. Each row has its own single byte DRB register. The value in a given DRB corresponds to the cumulative memory size including that row. For example, a value of 1 (0000 0001) in DRB0 (address lines 33 to 26) indicates that 64 Mbytes of DRAM has been populated in the first row.

DRB0 = Total memory in row0 (64 Mbyte increments)

DRB1 = Total memory in row0 + row1 (64 Mbyte increments)

DRB2 = Total memory in row0 + row1 + row2 (64 Mbyte increments)

DRB3 = Total memory in row0 + row1+ row2 + row3 (64 Mbyte increments)

The functionality of DRB3 is somewhat different than DRB[2:0]. In order to avoid a 64MByte “hole” at the top of memory, a value of 0x00 in DRB3, is interpreted as 0x100. In practice, this 0x00 value should not be set, since it implies addressing more memory than the EP80579 supported.

Note: The memory controller does not implement any hardware checks to prevent accesses to DRAM locations beyond what is populated in the DRB3 register. Such accesses are software programming errors and will result in unreliable operation.

Table 16-41 shows the DRBx to DIMM mapping. Please note that before populating the DIMM’s, all limitation described in Section 11.3.1, “Rules for Populating DIMM Slots” need to be followed.

Table 16-41. DRB to DIMM designation

	Even Row		Odd Row	
	Row / DRB Number	Address or DRB	Row / DRB Number	Address or DRB
DIMM0	DRB0	60h	DRB2	62h
DIMM1	DRB2	62h	NA	NA

Rules for programming the DRBx registers:

- DRB1 and DRB3 are unused and reserved.
  - — DRB1 and DRB3 should be programmed to be the same value as what was programmed in the even row. (DRB1 = DRB0, DRB3 = DRB2).
- Unpopulated rows must be programmed with a value of the last populated slot. This guarantees the unpopulated row will not be selected.
- Depending upon the configuration and amount of memory populated in each row, DRB0 and DRB2 should be programmed. This will correspond to on the EP80579, CS0# and CS1#.
  - — Single rank, 1 DIMM system programs DRB0 with the encoding for memory capacity in row 0. Further, DRB1 = DRB2 = DRB3 = DRB0.
  - — Single ranks in a 2 DIMM system programs DRB0 with the encoding for memory capacity in row 0 and DRB2 with the encoding for memory capacity in row 0 + row 1 + row 2. Further, DRB1 = DRB0 and DRB3 = DRB2.

— Single DIMM with dual ranks should program DRB0 with the encoding for memory capacity in row 0 and DRB2 with the encoding for memory capacity in row 0 + row 1 + row 2. Further, DRB1 = DRB0 and DRB3 = DRB2.

- It must always be the case with the EP80579 memory controller that for single rank, 1 DIMM systems, DRB0 = DRB1 = DRB2 = DRB3 and for all other configurations, DRB0 = DRB1 < DRB2 = DRB3.

Table 11-7, "Supported Rank Configurations in Single and Dual DIMM mode" on page 293 shows the mapping of the chip selects in the different legal rank populations on the EP80579.

Note: DRB0=DRAM Row 0, DRB1=DRAM Row 1, etc.

Table 16-42. Offset 60h: DRB[0-3] - DRAM Row [3:0] Boundary Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 60h at 1h **Offset End:** 60h at 1h	
Size: 8 bit	**Default:** ffh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	DRAM_RBA	**DRAM Row Boundary Address:** This 8 bit value defines the upper address for each row of DRAM rows. This 8 bit value is compared against a set of address lines to determine the upper address limit of a particular row. This field corresponds to bits 33:26of the system address.	N	ffh	RW

16.1.1.40 Offset 70h: DRA[0-1] – DRAM Row [0:1] Attribute Register

The DRAM Row Attribute register defines the DRAM technology. DRA is used to calculate the address mapping for column and row addresses as a function of DRAM technology specified in the DTYPE, DW and DIMMTECH fields.

- DRA0 describes characteristics of rows 0 (even) and 1 (odd).
- DRA1 describes characteristics of rows 2 (even) and 3 (odd).
- Due to the rules of the EP80579 DDR configuration, many fields of DRA0 and DRA1 are not meaningful (they exist only for backward compatibility). For most configurations, the values in DRA0[DW_EVEN] and DRA0[DIMMTECH_EVEN] are used. An exception is DRA1[DIMMTECH_EVEN] which is used to select the size of the devices on the second DIMM of a two DIMM system. See Table 16-43 for details.
- Bit fields that are not valid because the rank (or row) is not populated should not be changed. For such fields the reset value should be the benign state.
- The controller determines which of rows are populated after decoding the DRB registers. Please see Section 16.1.1.39, "Offset 60h: DRB[0-3] – DRAM Row [3:0] Boundary Register" for more details.

Note: All fields of the DRA[1:0] register need to be consistent or else unreliable operation will occur. For example the value programmed in NC_ODD should be consistent with the DIMMTECH_ODD field. The number of columns depends on the device technology.

Table 16-43. DRA[1:0] Field Selection

Case	DRA0 [DTYPE_ MSB]	Address	Value used for DIMMTECH	Value used for num_column / num_row	Value used for DW	Value used for DTYPE
1 DIMM, single rank	0 (single rank)	any address up to DRB0 limit	DRA0 [DIMMTECH_EVEN]	DRA0 [NC_EVEN/ NR_EVEN]	DRA0 [DW_EVEN]	DRA0 [DTYPE]
1 DIMM, dual rank	1 (dual rank)	any address up to DRB0 limit	DRA0 [DIMMTECH_EVEN]	DRA0 [NC_EVEN/ NR_EVEN]	DRA0 [DW_EVEN]	DRA0 [DTYPE]
	1 (dual rank)	any address between DRB0 and DRB2 limit	DRA0 [DIMMTECH_EVEN]	DRA0 [NC_EVEN/ NR_EVEN]	DRA0 [DW_EVEN]	DRA0 [DTYPE]
2 DIMMs, (one rank each)	0 (single rank per DIMM)	any address up to DRB0 limit	DRA0 [DIMMTECH_EVEN]	DRA0 [NC_EVEN/ NR_EVEN]	DRA0 [DW_EVEN]	DRA0 [DTYPE]
	0 (single rank per DIMM)	any address between DRB0 and DRB2 limit	DRA1 [DIMMTECH_EVEN]	DRA1 [NC_EVEN/ NR_EVEN]	DRA0 [DW_EVEN]	DRA0 [DTYPE]

Table 16-44. Offset 70h: DRA[0-1] - DRAM Row [0:1] Attribute Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 70h at 4h **Offset End:** 73h at 4h	
Size: 32 bit	**Default:** 00000515h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :29	NR_ODD	Number of Rows for odd numbered row. Functionality and encoding is exactly the same as NR_EVEN	N	000b	RW
28 :26	NC_ODD	Number of Columns for odd numbered row. Functionality and encoding is exactly the same as NC_EVEN	N	000b	RW
25 :23	NR_EVEN	Number of Rows for even numbered row: This information is used by the Mbist engine. Note that this field should be programmed to be consistent with the DIMMTECH fields of the DRA register. 000 — 8192 001 — 16,384 010 — 32,768 011 — 65,536 Others — Reserved	N	000b	RW

Table 16-44. Offset 70h: DRA[0-1] - DRAM Row [0:1] Attribute Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 70h at 4h **Offset End:** 73h at 4h	
Size: 32 bit	**Default:** 00000515h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
22 :20	NC_EVEN	Number of Columns for even numbered row. This information is used by the Mbist engine. 000 = 1024 001 = 2048 010 = 4096 011 = 8192 Others = Reserved Note that this field should be programmed to be consistent with the DIMMTECH fields of the DRA register.	N	000b	RW
19 :18	Reserved	Reserved	N	00b	RO
17 :14	DTYPE	**Device Type:** Bios sets these bits according to information read pertaining to the DIMMs installed bit 17: single or dual rank DIMM (0=single rank per DIMM) bit 16: DDR2 (0=DDR2) bit 15: 32 or 64 bit DDR populated (0=64bits) bit 14: unbuffered or registered DIMM (0=unbuffered)	N	0000b	RW
13 :12	Reserved	Reserved	N	00b	RO
11 :10	DWODD	**Device Width for odd-numbered row:** Functionality and encoding is exactly the same as DWEVEN. This value should be set to exactly the same as DWEVEN.	N	01b	RW
09 :06	DIMMTECH_ODD	**DIMM technology for odd-numbered row.** Functionality and encoding is exactly the same as DIMMTECH_EVEN (bits 3:0)	N	0100b	RW
05 :04	DWEVEN	**Device Width for even-numbered row:** BIOS sets this bit according to the width of the DDR2 SDRAM devices populated in this row. This is used to determine the page size and the DQS to DQ signal mapping. 00 = Reserved 01 = x8 DDR2 (1 strobe pair per byte) 10 = Reserved 11 = Reserved	N	01b	RW
03 :00	DIMMTECH_EVEN	**DIMM technology for even-numbered row:** BIOS sets this bit according to the density of the DDR devices populated in this row. This is used along with the device width and the DTYPE to determine the page size and the DQS to DQ signal mapping. 0000 = reserved 0011 = 2Gb DIMM 0100 = 1Gb DIMM 0101 = 512Mb DIMM 0110 = 256Mb DIMM others - reserved	N	0101b	RW

16.1.1.41 Offset 78h: DRT0 - DRAM Timing Register 0

The DRT register defines the DRAM timing parameter. For the EP80579, there are 2 DRT registers that need to be programmed based on the external capabilities of the memory devices, number of ranks/DIMM's, supported EP80579 memory configurations etc.

For details about the DRT1 register see "Offset 64h: DRT1 – DRAM Timing Register 1" on page 431.

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 1 of 7)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 78h **Offset End:** 7Bh	
Size: 32 bit	**Default:** 242AD280h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :29	BTBRWTA	**Back-To-Back Read-Write Turn Around:** This field determines the minimum number of CMDCLK on the DQ bus between Read-Write commands. It applies to RD-WR pairs to any destinations (in same or different rows). The purpose of this bit is to control the turnaround time on the DQ bus. The encoding below will be translated by the hardware into a number of CMDCLK's that will be inserted between read write commands.	N	001b	RW

Encoding	Command Clocks per Frequency
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 2 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td>Size: 32 bit</td><td colspan="3">Default: 242AD280h</td><td colspan="2">Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>28 :26</td><td>BTBRTA</td><td>Back To Back Read Turn Around: This field determines the minimum number of CMDCLK on the DQ bus between two reads destined to different ranks. The purpose of these bits is to control the turnaround time on the DQ bus.
The encoding below will be translated by the hardware into a number of CMDCLK's that will be inserted between read write commands.
<table><tr><td>Encoding</td><td>Command Clocks per Frequency</td></tr><tr><td>000</td><td>0</td></tr><tr><td>001</td><td>1</td></tr><tr><td>010</td><td>2</td></tr><tr><td>011</td><td>3</td></tr><tr><td>100</td><td>4</td></tr><tr><td>101</td><td>5</td></tr><tr><td>110</td><td>6</td></tr><tr><td>111</td><td>7</td></tr></table></td><td>N</td><td>001b</td><td>RW</td></tr>
<tr><td>25 :23</td><td>BBWRTA</td><td>Back to Back Write-Read turn around: This field determines the minimum number of CMDCLK on the DQ bus between Write-Read commands. The purpose of these 3 bits are to control the turnaround time on the DQ bus.
The encoding below will be translated by the hardware into a number of CMDCLK's that will be inserted between read write commands.
Command clocks apart based on the following encoding:
<table><tr><td>Encoding</td><td>Command Clocks per Frequency</td></tr><tr><td>000</td><td>0</td></tr><tr><td>001</td><td>1</td></tr><tr><td>010</td><td>2</td></tr><tr><td>011</td><td>3</td></tr><tr><td>100</td><td>4</td></tr><tr><td>101</td><td>5</td></tr><tr><td>110</td><td>6</td></tr><tr><td>111</td><td>7</td></tr></table></td><td>N</td><td>000b</td><td>RW</td></tr>
</table>

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 3 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 242AD280h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>22 :20</td><td>Trrd</td><td>Row Delay: The required row delay period between two activate commands accessing the same cs of a DIMM in tCK cycles.
JEDEC recommendation for this parameter is based on the device width. x8 devices = 7.5ns
<table>
<tr><th>Encoding</th><th>Number of CMDCLK delays</th></tr>
<tr><td>000</td><td>No delay</td></tr>
<tr><td>001</td><td>1</td></tr>
<tr><td>010</td><td>2</td></tr>
<tr><td>011</td><td>3</td></tr>
<tr><td>100</td><td>4</td></tr>
<tr><td>101</td><td>5</td></tr>
<tr><td>110</td><td>6</td></tr>
<tr><td>111</td><td>7</td></tr>
</table></td><td>N</td><td>010b</td><td>RW</td></tr>
<tr><td>19 :17</td><td>Twr</td><td>Write Recovery Delay: The required write recovery delay before being able to issue a precharge to the same page accessing the same cs/bank of a DIMM in tCK cycles.
JEDEC recommendation for this parameter is 15ns min.
<table>
<tr><th>Encoding</th><th>Number of CMDCLK delays</th></tr>
<tr><td>000</td><td>2</td></tr>
<tr><td>001</td><td>3</td></tr>
<tr><td>010</td><td>4</td></tr>
<tr><td>011</td><td>5</td></tr>
<tr><td>100</td><td>6</td></tr>
<tr><td>101</td><td>7</td></tr>
<tr><td>110</td><td>8</td></tr>
<tr><td>111</td><td>9</td></tr>
</table></td><td>N</td><td>101b</td><td>RW</td></tr>
</table>

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 4 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td>Size: 32 bit</td><td colspan="3">Default: 242AD280h</td><td colspan="2">Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>16 :12</td><td>Trc</td><td>This bit controls the number of DRAM clocks to enforce as the RAS cycle time.

<table>
<tr><th>Encoding</th><th>Number of CMDCLK delays</th></tr>
<tr><td>00000</td><td>11</td></tr>
<tr><td>00001</td><td>12</td></tr>
<tr><td>00010</td><td>13</td></tr>
<tr><td>00011</td><td>14</td></tr>
<tr><td>00100</td><td>15</td></tr>
<tr><td>00101</td><td>16</td></tr>
<tr><td>00110</td><td>17</td></tr>
<tr><td>00111</td><td>18</td></tr>
<tr><td>01000</td><td>19</td></tr>
<tr><td>01001</td><td>20</td></tr>
<tr><td>01010</td><td>21</td></tr>
<tr><td>01011</td><td>22</td></tr>
<tr><td>01100</td><td>23</td></tr>
<tr><td>01101</td><td>24</td></tr>
<tr><td>01110</td><td>25</td></tr>
<tr><td>01111</td><td>26</td></tr>
<tr><td>10000</td><td>27</td></tr>
<tr><td>10001</td><td>28</td></tr>
<tr><td>10010</td><td>29</td></tr>
<tr><td>Others</td><td>Reserved</td></tr>
</table></td><td>N</td><td>01101b</td><td>RW</td></tr>
</table>

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 5 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 242AD280h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>11 :09</td><td>Trcd</td><td>DRAM RAS# to CAS# delay: This bits controls the number of clocks inserted between a row activate command and a read or write command to that row
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>000</td><td>3</td></tr><tr><td>001</td><td>4</td></tr><tr><td>010</td><td>5</td></tr><tr><td>011</td><td>6</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>001b</td><td>RW</td></tr>
<tr><td>8 :06</td><td>Trp</td><td>DRAM RAS# Precharge: Time: the number of clock cycles needed to terminate access (precharge) to an open row of memory, and open access (activate) to the next row.
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>000</td><td>3</td></tr><tr><td>001</td><td>4</td></tr><tr><td>010</td><td>5</td></tr><tr><td>011</td><td>6</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>010b</td><td>RW</td></tr>
</table>

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 6 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0
Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 242AD280h</td><td colspan="2">Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>5 :03</td><td>CL</td><td>CAS# Latency: The number of clocks between the rising edge used by DRAMS to sample the Read Command and the rising edge that is used by the DRAM to drive read data.
DDR2 JEDEC Spec: Write latency (WL) is defined by a read latency (RL) minus one. Please refer to the DDR2 JEDEC Spec for more details.
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>000</td><td>3</td></tr><tr><td>001</td><td>4</td></tr><tr><td>010</td><td>5</td></tr><tr><td>011</td><td>6</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>000b</td><td>RW</td></tr>
</table>

Table 16-45. Offset 78h: DRT0 - DRAM Timing Register 0 (Sheet 7 of 7)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 78h
Offset End: 7Bh</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 242AD280h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>2 :00</td><td>PRGRPD</td><td>Programmable Read Pointer Delay: This bit field determines the read delay, which is based on both DIMM topology and technology.
The round trip timing budget has been estimated to be about 11.5 ns. Since an encoding of “000” means less than one command clock, the encoding values in this table refer to additional delays beyond one command clock.

Note that the PRGRPD encoding shown below is for 4 bits. The 4 bits are formed by concatenating DRT1[0] and DRT0[2:0].
Please refer to Section 16.1.1.42 for details on the DRT1 register.

<table>
<tr><td>PRGRPD[3:0] Encoding
DRT1[0], DRT0[2:0]</td><td>Number of CMDCLK delays</td></tr>
<tr><td>0000</td><td>0</td></tr>
<tr><td>0001</td><td>1</td></tr>
<tr><td>0010</td><td>2</td></tr>
<tr><td>0011</td><td>3</td></tr>
<tr><td>0100</td><td>4</td></tr>
<tr><td>0101</td><td>5</td></tr>
<tr><td>0110</td><td>6</td></tr>
<tr><td>0111</td><td>7</td></tr>
<tr><td>1000</td><td>8</td></tr>
<tr><td>1001</td><td>9</td></tr>
<tr><td>1010</td><td>10</td></tr>
<tr><td>Others</td><td>Reserved</td></tr>
</table>
</td><td>N</td><td>000b</td><td>RW</td></tr>
</table>

16.1.1.42 Offset 64h: DRT1 – DRAM Timing Register 1

This register controls the DRAM timing parameters.

For details about the DRT1 register see "Offset 78h: DRT0 - DRAM Timing Register 0" on page 424.

Table 16-46. Offset 64h: DRT1 - DRAM timing Register 1 (Sheet 1 of 4)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 64h
Offset End: 67h</td></tr>
<tr><td>Size: 32 bit</td><td colspan="2">Default: 12110000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>31 :28</td><td>tRAS</td><td>(Time for activation / RAS Active Strobe): time to activate a row of a bank (minimum time bank stays open before it can be closed/precharged again)
SW needs to program this parameter based on the DDR speed as shown in the table below.
Note however that the HW will use a different tRAS value when the DDR commands are generated by the Mbist Engine. During all other modes the controller will use the tRAS value programmed in this field.
<table><tr><th>DDR Speed</th><th>Encoding</th><th># of CMD clks</th><th>Tras value used by Mbist</th></tr><tr><td>400</td><td>0000</td><td>8 (40ns)</td><td>12</td></tr><tr><td>400</td><td>0001</td><td>9 (45ns)</td><td>12</td></tr><tr><td>533</td><td>0100</td><td>12 (45ns)</td><td>12</td></tr><tr><td>667</td><td>0111</td><td>15 (45ns)</td><td>15</td></tr><tr><td>800</td><td>1000</td><td>16 (40ns)</td><td>18</td></tr><tr><td>800</td><td>1010</td><td>18 (40ns)</td><td>18</td></tr></table></td><td>N</td><td>0001b</td><td>RW</td></tr>
<tr><td>27 :25</td><td>tRTP</td><td>RAS to Precharge (needed to calculate Read AutoPrecharge delay)
<table><tr><th>Encoding</th><th>Number of CMDCLK delays</th></tr><tr><td>000</td><td>2</td></tr><tr><td>001</td><td>3</td></tr><tr><td>010</td><td>4</td></tr><tr><td>011</td><td>5</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>001b</td><td>RW</td></tr>
</table>

Table 16-46. Offset 64h: DRT1 - DRAM timing Register 1 (Sheet 2 of 4)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 64h
Offset End: 67h</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 12110000h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>24 :20</td><td>tFAW</td><td>8 bank device Sequential Bank Activation Restriction: No more than 4 banks may be activated in a rolling tFAW window. Converting to clocks is done by dividing tFAW(ns) by tCK(ns) and rounding up to next integer value. As an example of the rolling window, if (tFAW/tCK) rounds up to 10 clocks, and an activate command.
This field is not valid for 4 banks device technologies like 256Mb x8 and 512 x8.
JEDEC recommendations:
1KB Page size = 37.5ns
2KB Page size = 50ns

<table><tr><th>Encoding</th><th>Number of CMDCLK delays</th></tr><tr><td>00000</td><td>No restriction</td></tr><tr><td>00001</td><td>7</td></tr><tr><td>00010</td><td>8</td></tr><tr><td>00011</td><td>9</td></tr><tr><td>00100</td><td>10</td></tr><tr><td>00101</td><td>11</td></tr><tr><td>00110</td><td>12</td></tr><tr><td>00111</td><td>13</td></tr><tr><td>01000</td><td>14</td></tr><tr><td>01001</td><td>15</td></tr><tr><td>01010</td><td>16</td></tr><tr><td>01011</td><td>17</td></tr><tr><td>01100</td><td>18</td></tr><tr><td>01101</td><td>19</td></tr><tr><td>01110</td><td>20</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>00001b</td><td>RW</td></tr>
</table>

Table 16-46. Offset 64h: DRT1 - DRAM timing Register 1 (Sheet 3 of 4)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 64h
Offset End: 67h</td></tr>
<tr><td>Size: 32 bit</td><td colspan="2">Default: 12110000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>19 :18</td><td>tCCD</td><td>CAS to CAS Delay
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>00</td><td>2</td></tr><tr><td>10</td><td>4</td></tr><tr><td>Others</td><td>Reserved</td></tr></table>Note: set tCCD to 2 clock delay for Burst of 4 (64 bit wide data interface) or set tCCD to 4 clock delay for Burst of 8 (32 bit wide data interface).</td><td>N</td><td>00b</td><td>RW</td></tr>
<tr><td>17 :15</td><td>tWTR</td><td>Internal Write to Read command delay, at least 2 x tCK and independent of operating frequency
JEDEC recommendations for DDR2
400MTS = 10ns
Others = 7.5ns
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>000</td><td>No Delay</td></tr><tr><td>001</td><td>1</td></tr><tr><td>010</td><td>2</td></tr><tr><td>011</td><td>3</td></tr><tr><td>100</td><td>4</td></tr><tr><td>101</td><td>5</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>N</td><td>010b</td><td>RW</td></tr>
<tr><td>14 :13</td><td>BLEN</td><td>Burst length
<table><tr><td>Encoding</td><td>Burst Length</td></tr><tr><td>00</td><td>4
(DDR2 64 bit data width)</td></tr><tr><td>01</td><td>8
(DDR2 32 bit data width)</td></tr></table></td><td>N</td><td>00b</td><td>RW</td></tr>
</table>

Table 16-46. Offset 64h: DRT1 - DRAM timing Register 1 (Sheet 4 of 4)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 64h
Offset End: 67h</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 12110000h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>12 :12</td><td>2Tor1T</td><td>2T or 1T timing on the command bus to DRAM devices.
<table><tr><td>Encoding</td><td>Timing</td></tr><tr><td>0</td><td>1T</td></tr><tr><td>1</td><td>2T</td></tr></table></td><td>N</td><td>0b</td><td>RW</td></tr>
<tr><td>11 :04</td><td>NOPCNT</td><td>Programmable NOP insertion:
Number of NOPs will be inserted between read/write commands to slow down Membist activities in the same page.
Up to 255 clocks NOPs can be programmed to insert delay between read/write commands. If NOPs delay is programmable less than the required DRAM timing, Overall NOP delay from command to command will not be seen.</td><td>N</td><td>00h</td><td>RW</td></tr>
<tr><td>03 :01</td><td>BTBWTA</td><td>Back-To-Back Write Turn Around: This field determines the data bubble duration between write data bursts. It applies to WR-WR pairs to different ranks, and is only expected to be used in DDR2 mode with ODT enabled in the event that ODT selections must change between ranks. The purpose of this field is to control the data burst spacing on the DQ bus.
The encoding below will be translated by the hardware into a number of CMDCLK's that will be inserted between read write commands.
<table><tr><td>Encoding</td><td>Number of CMDCLK delays</td></tr><tr><td>000</td><td>0</td></tr><tr><td>001</td><td>1</td></tr><tr><td>010</td><td>2</td></tr><tr><td>011</td><td>3</td></tr><tr><td>100</td><td>4</td></tr><tr><td>101</td><td>5</td></tr><tr><td>110</td><td>6</td></tr><tr><td>111</td><td>7</td></tr></table></td><td>N</td><td>000b</td><td>RW</td></tr>
<tr><td>0 :0</td><td>PRGRPD_4</td><td>Bit[3] of the Programmable Read Pointer Delay field. Please refer to DRT0[2:0] for more details on this bit field (Section 16.1.1.41).</td><td>N</td><td>0b</td><td>RW</td></tr>
</table>

16.1.1.43 Offset 7Ch: DRC – DRAM Controller Mode Register

This register controls the mode of the DRAM controller.

Table 16-47. Offset 7Ch: DRC - DRAM Controller Mode Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 7Ch **Offset End:** 7Fh	
Size: 32 bit	**Default:** 00000002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :31	CKE1	This bit controls the value that will be driven on the CKE[1] pin. Note however that CKEPNM bit in this CSR can override and force low the state of both CKE[1:0] pins regardless of the state of this bit. BIOS can write to this CSR bit to directly control the state of the CKE pin. HW will update the state of this bit based on self-refresh exit command. 0 Drive CKE[1] low, de-activates DRAM devices 1 Drive CKE[1] asserted, activates DRAM devices	Y	0b	RW
30 30	CKE0	This bit controls the value that will be driven on the CKE[0] pin. Note however that CKEPNM bit in this CSR can override and force low the state of both CKE[1:0] pins regardless of the state of this bit. BIOS can write to this CSR bit to directly control the state of the CKE pin. HW will update the state of this bit based on self-refresh exit command. 0 Drive CKE[0] low, de-activates DRAM devices 1 Drive CKE[0] asserted, activates DRAM devices	Y	0b	RW
29 29	IC	0 = **Initialization Complete:** This bit is used for communication of software state between the memory controller and the BIOS. DRAM interface has not been initialized. 1 = DRAM interface has been initialized.	N	0b	RW
28 :22	Reserved	Reserved	N	0000000b	RO
21 :20	DDIM	DRAM Data Integrity Mode: These bits select DRAM data integrity modes. When in non-ECC mode no ECC correction is done and no ECC errors are logged in the FERR/NERR registers. 00 Non-ECC mode 01 ECC enabled 10 Reserved 11 Reserved	N	00b	RW
19 :14	Reserved	Reserved	N	0b	RO
13	HLDDIS	Command/address hold disable Disabling hold will allow the address and bank address pins to revert to all zeros during idle cycles. When hlddis is clear, the addresses retain the value of the last non-idle command cycle in order to reduce switching on the bus. 0 = disabled, 1 enabled	Y	0b	RW
12	CADIS	DDR Command/address pin output disable: This bit controls Address, bank address, CAS, RAS, WE. 0 = Enabled 1 = Disabled	Y	0b	RW

Table 16-47. Offset 7Ch: DRC - DRAM Controller Mode Register (Sheet 2 of 2)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: Configuration</td><td>Bus:Device:Function: 0:0:0</td><td>Offset Start: 7Ch
Offset End: 7Fh</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 00000002h</td><td></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>11 :10</td><td>CSDIS</td><td>DDR Chip select output disable
bit[11] = CS[1]
bit[10] = CS[0]
0 = Enabled
1 = Disabled</td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>9 9</td><td>Reserved</td><td>Reserved</td><td>N</td><td>0b</td><td>RO</td></tr>
<tr><td>7 8</td><td>Reserved_2</td><td>Reserved_RW_Sticky</td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>06 :05</td><td>ODTDIS</td><td>DRAM ODT Disable: bit[5] = ODT[0]
bit[6] = ODT[1]
0 = Enables the use of ODT when running
1 = Disables (tristates) the use of ODT when running</td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>04 04</td><td>CKEPNM</td><td>CKE pin mode:
0 = Force low. Forces the state of CKE[1:0] low. When this bit is cleared it over-rides all functionality that drives the CKE pin and forces it low. SW needs to set this bit for normal operation
1 = Enable CKE[1:0]. BIOS will set this bit to a 1 for normal operating mode.</td><td>Y</td><td>0b</td><td>RW</td></tr>
<tr><td>3 :0</td><td>DS</td><td>The PLL only supports one update of ratio (the lower nibble of this register). This register defaults to DDR2-400
This field reflects BIOS selection of DDR speed, which may have been “down-binned” due to fuse settings (see SDRC.FUSESPEED)
<table><tr><td>Encoding</td><td>DDR Data Speed (MT/s)</td><td>DDR CMD Freq (Mhz)</td><td></td></tr><tr><td>0x10</td><td>400</td><td>200</td><td>Default</td></tr><tr><td>0x00</td><td>533</td><td>266</td><td></td></tr><tr><td>0111</td><td>667</td><td>333</td><td></td></tr><tr><td>0101</td><td>800</td><td>400</td><td></td></tr><tr><td>Others</td><td>Reserved</td><td></td><td></td></tr></table></td><td>Y</td><td>0010b</td><td>RWO</td></tr>
</table>

16.1.1.44 Offset 84h: ECCDIAG – ECC Detection/Correction Diagnostic Register

The MEMPEN bit of this register controls if the memory controller poisons write data to the DRAM when it detects a parity error on its interface. By default, this bit is clear and the memory controller will not poison the write data to DRAM when it detects a parity error on its interface. If MEMPEN is set, and a write transaction with bad parity is sent to memory controller from either IMCH or AIOC, the write to DDR will be poisoned as follows: each bit of ECC will be flipped from the value otherwise calculated for that data write, based on the “bad” data sent from either IMCH or AIOC.

Table 16-48. Offset 84h: ECCDIAG - ECC Detection/Correction Diagnostic Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :21	Reserved	Reserved	N	000b	RO
20 20	FECCDT	**Flip ECC on all data transfers:** Flip the designated ECC bits (bits 15:00) on all data transfers to DRAM. If a cacheline is in progress when this register is written, wait until the start of the next cacheline to flip parity bits. Note that if FECCDT and MEMPEN is set and a bad parity is detected, the M_unit will poison and flip the ECC bits.	N	0b	RW
19 19	Reserved	Reserved	N	0b	RO

Table 16-48. Offset 84h: ECCDIAG - ECC Detection/Correction Diagnostic Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
18 18	MEMPEN	**Memory Poison Enable:** Allows for propagation of data errors not initiated by this feature to DRAM. Error injection via bit 20 is possible regardless of this bit setting. The setting of this bit has no effect on the reporting or logging of data errors. 0 = Error poisoning is disabled, data errors are not propagated, meaning that only good ECC is generated ECC mode even when bad parity is detected on its interface. 1 = Error poisoning enabled when in ECC mode. The memory controller will poison the write data to DRAM when it detects a parity error on its interface.	N	0b	RW
17 :16	DPRSL	**Data pair selector:** This two-bit field selects which pair of quad-words in a cache line the inversion vector is applied against. Regardless of what operational mode the memory subsystem is in, this field always applies to the same QW pair. QW0 corresponds to data bits 63:00, QW1 to [127:64] … and QW7 corresponds to [511:448] 00 QW0 and QW1 01 QW2 and QW3 10 QW4 and QW5 11 QW6 and QW7	N	00b	RW
15 : 00	ECCBIN	**ECC bit invert vector:** This vector operates individually for every ECC bit in the selected High or Low ECC block, during writes to DRAM. For all k between 0 and 15, when bit (k) set to 1, the value of the k ECC bit (which corresponds with the k data byte lane) is inverted. Otherwise, the value the k ECC bit is not affected. In other words, bits 15:08 are applied to the ECC vector of the high Qword in the selected pair, and bits 07:00 are applied to the ECC vector of the low Qword in the selected pair. For Example: Data Pair Selector bits 17:16 = 00 ECC bit invert vector bits 15:08 are applied to the ECC vector for QW1 ECC bit invert vector bits 07:00 are applied to the ECC vector for QW0	N	0000h	RW

16.1.1.45 Offset 88h: SDRC – DDR SDRAM Secondary Control Register

This register is used or setting memory controller parameters such as queue depths, scheduler parameters, arbiter parameters, AIOC and IA-32 core stream enabling, IA-32 core parity checking, bank remapping, etc.

Table 16-49. Offset 88h: SDRC - DDR SDRAM Secondary Control Register (Sheet 1 of 3)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 88h
Offset End: 8Bh</td></tr>
<tr><td>Size: 32 bit</td><td colspan="2">Default: 00000002h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>31 :30</td><td>ODTZENA1</td><td>On Die Termination Enable: These bits enable the EP80579 on die termination. ODT control for the DQ[71:64]/DQS[8] (ECC byte) buffers on the inbound read path.
<table><tr><td>Encoding</td><td>ODT</td></tr><tr><td>00</td><td>Disabled</td></tr><tr><td>01</td><td>60 ohms</td></tr><tr><td>10</td><td>120 ohms</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>29 :28</td><td>ODTZENA</td><td>On Die Termination Enable: These bits enable the EP80579 on die termination. ODT for the DQ[63:0]/DQS[7:0] buffers on the inbound read path
<table><tr><td>Encoding</td><td>ODT</td></tr><tr><td>00</td><td>Disabled</td></tr><tr><td>01</td><td>60 ohms</td></tr><tr><td>10</td><td>120 ohms</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>27 :26</td><td>Reserved</td><td>Reserved</td><td>Y</td><td>00b</td><td></td></tr>
<tr><td>25 22</td><td>Reserved</td><td>Reserved</td><td>N</td><td>0x0b</td><td>RW</td></tr>
<tr><td>21 :20</td><td>FUSESPEED</td><td>Fuse Speed - Read only copy of the DDR speed fuse setting
00b - DDR-800 MTS
01b - DDR-667 MTS
10b - DDR-533 MTS
11b - DDR-400 MTS</td><td>N</td><td>fuse</td><td>RO</td></tr>
</table>

Table 16-49. Offset 88h: SDRC - DDR SDRAM Secondary Control Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 88h **Offset End:** 8Bh	
Size: 32 bit	**Default:** 00000002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
19 :17	DRRIRR	Demand Scrub Retry (DRR) Injection Rate Regulator: This field determines the minimum rate at which DRRs will be scheduled to the DRAM. If multiple DRR's are pending in the MC, they will be issued to the DRAM spaced apart by at least DRRIRR by the hardware. 000b - 4 DDR controller clock cycles (Default) Others - Reserved The only supported/validated value is 000b. ***Note:*** This feature should not be confused with DED retry feature that is not support by the EP80579.	N	000b	RW
16 16	DDRDIS	Demand Scrub Retry (DRR) Disable: This bit by default is set to 0 to enable the demand scrub retry feature. When enabled any demand scrub writes that do not get scheduled to DRAM will be retried. When this bit is set, demand scrubs that are dropped will not be retried. ***Note:*** This feature should not be confused with DED retry feature that is not support by the EP80579.	N	0b	RW
15 :12	SCH_WGT	Controls for weighted round robin scheduling when both IA and AIOC requests are posted for accessing the same bank. Selects the number of IA transfers consecutively selected instead of AIOC transfers when requests are posted from both, and from the same bank. Works in conjunction with eight 2 bit counter, one for each bank, which indicates how many IA transfers, from each bank, have been selected since the last AIOC transfer was selected form that bank. xx00 = choose AIOC command if AIOC and IA are both present, and the last 1 command selected was IA. If no IA commands are present, choose AIOC and reset 2 bit bank count of IA transfers for the selected bank. xx01 = choose AIOC command if AIOC and IA are both present, and the last 2 commands selected were IA. If no IA commands are present, choose AIOC and reset 2 bit bank count of IA transfers for the selected bank xx11 = choose AIOC command if AIOC and IA are both present, and the last 3 commands selected were IA. If no IA commands are present, choose AIOC and reset 2 bit bank count of IA transfers for the selected bank Otherwise = reserved	N	0000b	RW
11	mu_enable_aioccmd	Enable scheduler to pass AIOC transfers to DDR	N	0b	RW
10	mu_enable_bcmd	Enable scheduler to pass IMCH transfers to DDR	N	0b	RW
09	mu_enable_eccrrwcmd	Enable scheduler to pass internally generated demand scrubs (upon detection of single bit ECC error) transfers to DDR. Please also refer SDRC.DDRDIS for the Demand Scrub Retry Feature. In order to ensure that no demand scrubs are dropped, the DRR feature should be enabled.	N	0b	RW
08	mu_enable_bscrubcmd	Enable scheduler to pass internally generated background scrub transfers to DDR	N	0b	RW

Table 16-49. Offset 88h: SDRC - DDR SDRAM Secondary Control Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 88h **Offset End:** 8Bh	
Size: 32 bit	**Default:** 00000002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	ASU_CMDQSIZE	Size of command queue available to ASU The Scheduler implements a shared Command Queue which is nominally 64 entries deep. This queue is shared between ASU and IA traffic. In order to reserve some queue entries for IA commands only, software is able to set an upper limit on the number of ASU commands that can occupy this queue. If the number of ASU commands exceeds this programmed value, subsequent AIOC commands may be backed off by the memory controller, until ASU commands drain to DDR, and the number in the command queue, once again, falls below the programmed value. Recommended value is 0x30. Do not set this field to 0.	N	00000010b	RW

16.1.1.46 Offset 8Ch: CKDIS – CK/CK# Clock Disable Register

This register is used to enable or disable the CK/CK# pins to the DIMMS. This feature is intended to reduce EMI and power consumption due to clocks toggling to DIMMs that are not populated.

Table 16-50. Offset 8Ch: CKDIS - CK/CK# Clock Disable Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:0	**Offset Start:** 8Ch **Offset End:** 8Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7 :06	Reserved	Reserved	N	00b	RO
05 :00	CKDIS	CK/CK# Disable (Sticky) Each bit corresponds to a pair of CK pins. Default is enabled. 0 = Enable CK signals 1 = Disable CK signals. When disabled, the CK/CK# signals are tristated. Bit 5: Enable/Disable CK[5]/CK#[5] Bit 4: Enable/Disable CK[4]/CK#[4] Bit 3: Enable/Disable CK[3]/CK#[3] Bit 2: Enable/Disable CK[2]/CK#[2] Bit 1: Enable/Disable CK[1]/CK#[1] Bit 0: Enable/Disable CK[0]/CK#[0]	Y	000000b	RW

16.1.1.47 Offset 8Dh: CKEDIS - CKE Clock Disable Register

This register is used to enable or disable the CKE pins to the DIMMS.

Also see Section 16.1.1.43, "Offset 7Ch: DRC – DRAM Controller Mode Register" CSR that describes the CKEPNM bit that can force the CKE[1:0] pins low.

Table 16-51. Offset 8Dh: CKEDIS - CKE Clock Enable Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 8Dh **Offset End:** 8Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7 :03	Reserved	Reserved	N	00000b	RO
02 02	CKE1DIS	CKE1 Disable (Sticky) Bit corresponds CKE[1] pin. Default is enabled. 1 = Disable CKE[1] signals 0 = Enable CKE[1] signals. When disabled, the CKE[1] pin is tristated.	Y	0b	RW
01 01	Reserved	Reserved	N	0b	RO
00 00	CKE0DIS	CKE0 Disable (Sticky) Bit controls CKE[0] pin. Default is enabled. 1 = Disable CKE[0] signals 0 = Enable CKE[0] signals. When disabled, the CKE[0] pin is tristated.	Y	0b	RW

16.1.1.48 Offset 90h: SPARECTL - SPARE Control Register

This register is used to set the prescale values for the leaky bucket error counting mechanism.

Table 16-52. Offset 90h: SPARECTL - SPARE Control Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** 90h **Offset End:** 93h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	DEDEPV	**DED error prescale value:** Prescale value ranges from 0-255.	Y	00h	RW
23 :16	SECEPV	**SEC error prescale value:** Prescale value ranges from 0-255.	Y	00h	RW
15 :12	DEDEPU	**DED error prescale unit:** 0000 Never 0001 1 µs 0010 1 ms 0011 1 s 0100 1 minute 0101 1 hour 0110 1 day 0111 1 week 1XXX Never	Y	0000b	RW
11 :08	SECEPU	**SEC error prescale unit:** 0000 Never 0001 1 µs 0010 1 ms 0011 1 s 0100 1 minute 0101 1 hour 0110 1 day 0111 1 week 1XXX Never	Y	0000b	RW
07 :00	Reserved	Reserved	N	00h	RO

16.1.1.49 Offset B0h: DDR2ODTC - DDR2 ODT Control Register

The DDR2ODTC controls the behavior of the ODT[1:0] output pins. The ODT pins control the on-die termination on the DDR DRAM devices. (To control the ODT behavior on the EP80579 please refer to Section 11.4.3, “On-Die Termination (ODTZ) on the EP80579”.)

The DDR2ODTC control register provides separate fields to control the ODT pin behavior for each of the four possible active cases: read to rank0, read to rank1, write to rank0, and write to rank1. Each field is two bits wide as there are two ODT pins on the EP80579. Please refer the different ODT configurations that the EP80579 supports as shown in Figure 11-4, “ODT Timing on Back-to-Back Reads to Different Slots” on page 300 and Figure 11-5, “ODT Timing on Back-to-Back Writes to Different Slots” on page 301.

Table 16-53. Offset B0h: DDR2ODTC - DDR2 ODT Control Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:0		**Offset Start:** B0h **Offset End:** B3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :14	Reserved	Reserved	N	0000h	RO
13 12	R1ODTWR	R1ODTWR: Value for logical ODT[1:0] for the case of write access to logical rank 1.	N	00b	RW
11 10	Reserved	Reserved	N	00b	RO
9 8	R1ODTRD	R1ODTRD:Value for logical ODT[1:0] for the case of read access to logical rank 1.	N	00b	RW
7 6	Reserved	Reserved	N	00b	RO
5 4	R0ODTWR	R0ODTWR. Value for logical ODT[1:0] for the case of write access to logical rank 0.	N	00b	RW
3 2	Reserved		N	00b	RO
1 0	R0ODTRD	R0ODTRD. Value for logical ODT[1:0] for the case of read access to logical rank 0.	N	00b	RW

16.2 DRAM Controller Error Reporting Registers: Bus 0, Device 0, Function 1

The DRAM Controller Error Reporting registers are in Bus 0, Device 0, Function 1. Table 16-54 provides the register address map for this device and function.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 16-54. Bus 0, Device 0, Function 1: Summary of IMCH Error Reporting PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 447	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 447	5021h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 448	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 448	0000h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 449	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 449	00h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 449	FFh
0Dh	0Dh	"Offset 0Dh: MLT - Master Latency Timer Register" on page 450	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 450	00h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 450	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 451	0000h
40h	43h	"Offset 40h: GLOBAL_FERR - Global First Error Register" on page 451	00000000h
44h	47h	"Offset 44h: GLOBAL_NERR - Global Next Error Register" on page 453	00000000h
48h	4Bh	"Offset 48h: NSI_FERR - NSI First Error Register" on page 454	00000000h
4Ch	4Fh	"Offset 4Ch: NSI_NERR - NSI Next Error Register" on page 457	00000000h
50h	53h	"Offset 50h: NSI_SCICMD - NSI SCI Command Register" on page 459	00000000h
54h	57h	"Offset 54h: NSI_SMICMD: NSI SMI Command Register" on page 461	00000000h
58h	5Bh	"Offset 58h: NSI_SERRCMD - NSI SERR Command Register" on page 464	00000000h
5Ch	5Fh	"Offset 5Ch: NSI_MCERRCMD - NSI MCERR Command Register" on page 466	00000000h
60h	61h	"Offset 60h: FSB_FERR - FSB First Error Register" on page 468	0000h
62h	63h	"Offset 62h: FSB_NERR - FSB Next Error Register" on page 469	0000h
64h	65h	"Offset 64h: FSB_EMASK - FSB Error Mask Register" on page 470	0009h
68h	69h	"offset 68h: FSB_SCICMD - FSB SCI Command Register" on page 471	0000h
6Ah	6Bh	"Offset 6Ah: FSB_SMICMD - FSB SMI Command Register" on page 472	0000h
6Ch	6Dh	"Offset 6Ch: FSB_SERRCMD - FSB SERR Command Register" on page 473	0000h
6Eh	6Fh	"Offset 6Eh: FSB_MCERRCMD - FSB MCERR Command Register" on page 474	0000h
70h	70h	"Offset 70h: BUF_FERR - Memory Buffer First Error Register" on page 475	00h
72h	72h	"Offset 72h: BUF_NERR - Memory Buffer Next Error Register" on page 475	00h
74h	74h	"Offset 74h: BUF_EMASK - Memory Buffer Error Mask Register" on page 476	00h

Table 16-54. Bus 0, Device 0, Function 1: Summary of IMCH Error Reporting PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
78h	78h	"Offset 78h: BUF_SCICMD - Memory Buffer SCI Command Register" on page 477	00h
7Ah	7Ah	"Offset 7Ah: BUF_SMICMD - Memory Buffer SMI Command Register" on page 478	00h
7Ch	7Ch	"Offset 7Ch: BUF_SERRCMD - Memory Buffer SERR Command Register" on page 479	00h
7Eh	7Eh	"Offset 7Eh: BUF_MCERRCMD - Memory Buffer MCERR Command Register" on page 480	00h
E4h	E7h	"Offset E4h: NSIERRINJCTL - NSI Error Injection Control Register" on page 481	00040000h
E8h	EBh	"Offset E8h: BERRINJCTL - Buffer Error Injection Control Register" on page 482	00000000h
80h	81h	"Offset 80h: DRAM_FERR - DRAM First Error Register" on page 483	0000h
82h	83h	"Offset 82h: DRAM_NERR - DRAM Next Error Register" on page 484	0000h
84h	84h	"Offset 84h: DRAM_EMASK - DRAM Error Mask Register" on page 486	00h
88h	88h	"Offset 88h: DRAM_SCICMD - DRAM SCI Command Register" on page 487	00h
8Ah	8Ah	"Offset 8Ah: DRAM_SMICMD - DRAM SMI Command Register" on page 488	00h
8Ch	8Ch	"Offset 8Ch: DRAM_SERRCMD - DRAM SERR Command Register" on page 489	00h
8Eh	8Eh	"Offset 8Eh: DRAM_MCERRCMD - DRAM MCERR Command Register" on page 490	00h
98h	99h	"Offset 98h: THRESH_SEC0 - Rank 0 SEC Error Threshold Register" on page 491	0000h
9Ah	9Bh	"Offset 9Ah: THRESH_SEC1 - Rank 1 SEC Error Threshold Register" on page 491	0000h
A0h	A3h	"Offset A0h: DRAM_SECF_ADD - DRAM First Single Bit Error Correct Address Register" on page 492	00000000h
A4h	A7h	"Offset A4h: DRAM_DED_ADD - DRAM Double Bit Error Address Register" on page 492	00000000h
A8h	ABh	"Offset A8h: DRAM_SCRB_ADD - DRAM Scrub Error Address Register" on page 493	00000000h
B0h	B1h	"Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register" on page 494	0000h
B2h	B3h	"Offset B2h: DRAM_DED_R0 - DRAM Rank 0 DED Error Counter Register" on page 494	0000h
B4h	B5h	"Offset B4h: DRAM_SEC_R1 - DRAM Rank 1 SEC Error Counter Register" on page 494	0000h
B6h	B7h	"Offset B6h: DRAM_DED_R1 - DRAM Rank 1 DED Error Counter Register" on page 495	0000h
C2h	C3h	"Offset C2h: THRESH_DED - DED Error Threshold Register" on page 495	0000h
C4h	C5h	"Offset C4h: DRAM_SECF_SYNDROME - DRAM First Single Error Correct Syndrome Register" on page 496	0000h
C6h	C7h	"Offset C6h: DRAM_SECN_SYNDROME - DRAM Next Single Error Correct Syndrome Register" on page 496	0000h
C8h	CBh	"Offset C8h: DRAM_SECN_ADD - DRAM Next Single Bit Error Correct Address Register" on page 497	00000000h
DCh	DDh	"Offset DCh: RANKTHREX - Rank Error Threshold Exceeded Register" on page 498	0000h
ECh	EFh	"Offset ECh: DERRINJCTL - DRAM Error Injection Control Register" on page 499	00000000h

16.2.1 Register Details

16.2.1.1 Offset 00h: VID - Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device.

Table 16-55. Offset 00h: VID - Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:**	8086h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel 8086h.		8086h	RO

16.2.1.2 Offset 02h: DID - Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device.

Table 16-56. Offset 02h: DID - Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:**	5021h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IMCH Host-Bridge Function 1.		5021h	RO

16.2.1.3 Offset 04h: PCICMD - PCI Command Register

Since IMCH Device 0 does not physically reside on PCI_A (internal bus) many of the bits are not implemented.

Table 16-57. Offset 04h: PCICMD - PCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		00h	
08	SERRE	**SERR Enable:** This bit is a global enable bit for Device 0 SERR messaging. 0 = Disable, SERR message is not generated by the IMCH for Device 0, Function 1. 1 = The IMCH is enabled to generate SERR messages over the NSI for specific Device 0, Function 1 error conditions that are individually enabled in the ERRCMD registers. The IMCH communicates the SERR condition by sending a DO_SERR message over NSI to the IICH. If this bit is set to a 1, the IMCH is enabled to generate SERR messages over NSI for specific Device 0, Function 1 error conditions that are individually enabled in the NSI_SERRCMD, FSB_SERRCMD, BUF_SERRCMD, and DRAM_SERRCMD registers. The error status is reported in the PCISTS register as well as the corresponding FERR/NERR registers. ***Note:*** Reporting via SERR for detected parity error which is essentially NSI Poisoned TLP's, can ALSO be reported through by the Device 0, Function 0.		0b	RW
07 : 00	Reserved	Reserved		0b	

16.2.1.4 Offset 06h: PCISTS - PCI Status Register

PCISTS is a 16-bit status register that reports the occurrence of error events on Device 0's PCI interface.

Table 16-58. Offset 06h: PCISTS - PCI Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	Reserved	Reserved		0b	
14	SSE	**Signaled System Error:** 0 = SERR is not generated by IMCH Device 0, Function 1 1 = IMCH Device 0 Function 1 generated a SERR message over NSI for any enabled Device 0, Function 1 error condition. Software clears this bit by writing a 1 to the bit location.		0b	RWC
13 : 00	Reserved	Reserved		000h	

16.2.1.5 Offset 08h: RID - Revision Identification Register

This register contains the revision number of the IMCH Device 0.

Table 16-59. Offset 08h: RID - Revision Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:**	Variable		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the IMCH Device 0.		Variable	RO

16.2.1.6 Offset 0Ah: SUBC - Sub-Class Code Register

Table 16-60. Offset 0Ah: SUBC - Sub-Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SUBC	**Sub-Class Code:** This value indicates the Sub Class Code into which the IMCH falls. 00h = Bridge		00h	RO

16.2.1.7 Offset 0Bh: BCC - Base Class Code Register

Table 16-61. Offset 0Bh: BCC - Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:**	FFh		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BASEC	**Base Class Code:** This value indicates the Base Class Code for the IMCH Device 0, Function 1. FFh = A 'non-defined' device. Since this function is used for error conditions, it does not fall into any other class.		FFh	RO

16.2.1.8 Offset 0Dh: MLT - Master Latency Timer Register

Device 0 in the IMCH is not a PCI master so this register is not implemented.

Table 16-62. Offset 0Dh: MLT - Master Latency Timer Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	Reserved	Reserved			00h	

16.2.1.9 Offset 0Eh: HDR - Header Type Register

Table 16-63. Offset 0Eh: HDR - Header Type Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header:** This value indicates the Header Type for the IMCH Device 0. 00h = IMCH is a multi-function device with a standard header layout.			00h	RO

16.2.1.10 Offset 2Ch: SVID - Subsystem Vendor Identification Register

This value is used to identify the vendor of the subsystem.

Table 16-64. Offset 2Ch: SVID - Subsystem Vendor Identification Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBVID	**Subsystem Vendor ID:** This field must be programmed during boot-up to indicate the vendor of the system board.			0000h	RWO

16.2.1.11 Offset 2Eh: SID - Subsystem Identification Register

This value is used to identify a particular subsystem.

Table 16-65. Offset 2Eh: SID - Subsystem Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0000h	RWO

16.2.1.12 Offset 40h: GLOBAL_FERR - Global First Error Register

This register is used to log various error conditions at the “unit” level. These bits are “sticky” through reset, and are set regardless of whether or not any error messages (SCI, SMI, SERR#, MCERR#) are enabled and generated at the unit level. Specific error conditions within the various functional units are logged in the unit-specific error registers that follow.

This register captures the FIRST global Fatal and the FIRST global Non-Fatal errors. For these global error registers, a non-fatal error can be either an uncorrectable error which is non-fatal, or a correctable error. Any future errors (NEXT errors) are captured in the Global_NERR register. No further error bits in this register are set until the existing error bit is cleared.

Note: If multiple errors are reported in the same clock as the first error, all errors are latched.

Table 16-66. Offset 40h: GLOBAL_FERR - Global First Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	Reserved	Reserved		0b	
27	DRAM_FE	**DRAM Controller Channel Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal DRAM I/F error. 1 = The IMCH detected a fatal DRAM interface error.	Y	0b	RWC
26	FSB_FE	**Host (FSB) Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal FSB error. 1 = The IMCH detected a fatal FSB error.	Y	0b	RWC
25	NSI_FE	NSI **Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal NSI error. 1 = The IMCH detected a fatal NSI error.	Y	0b	RWC

Table 16-66. Offset 40h: GLOBAL_FERR - Global First Error Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
24	DMA_FE	**DMA Controller Fatal Error Device 1 fatal error (EDMA):** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal DMA Controller error. 1 = The IMCH detected a fatal DMA controller error.	Y	0b	
23	PA_FE	**PCI Express* Port A(0) Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal PCI Express Port A error. 1 = The IMCH detected a fatal PCI Express Port A(0) error.	Y	0b	RWC
22	PA1_FE	**PCI Express Port A1 Fatal Error** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal PCI Express Port A1 error. 1 = The IMCH detected a fatal PCI Express Port A1 error.	Y	0b	RWC
21 : 15	Reserved	Reserved		0b	
14	BUFF_NFE	**Buffer unit detected non-fatal error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal Buffer error. 1 = The IMCH detected a non-fatal Buffer error.	Y	0b	RWC
13	DRAM_NFE	**DRAM Controller Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal DRAM Controller error. 1 = The IMCH detected a non-fatal DRAM Controller error.	Y	0b	RWC
12	FSB_NFE	**FSB Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal FSB error. 1 = The IMCH detected a non-fatal FSB error.	Y	0b	RWC
11	NSI_NFE	NSI **Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal NSI error. 1 = The IMCH detected a non-fatal NSI error.	Y	0b	RWC
10	DMA_NFE	**DMA Controller Non-Fatal Error Device 1 non-fatal error (EDMA):** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal DMA Controller error. 1 = The IMCH detected a non-fatal DMA Controller error.	Y	0b	
09	PA_NFE	**PCI Express Port A(0) Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal PCI Express Port A error. 1 = The IMCH detected a non-fatal PCI Express Port A error.	Y	0b	RWC
08	PA1_NFE	**PCI Express Port A1 Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal PCI Express Port A1 error. 1 = The IMCH detected a non-fatal PCI Express Port A1 error.	Y	0b	RWC
07 : 00	Reserved	Reserved		00h	

16.2.1.13 Offset 44h: GLOBAL_NERR - Global Next Error Register

The bit definitions are defined for GLOBAL_FERR.

Table 16-67. Offset 44h: GLOBAL_NERR - Global Next Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 44h **Offset End:** 47h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	Reserved	Reserved		0b	
27	DRAM_FE	**DRAM Controller Channel Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal DRAM I/F error. 1 = The IMCH detected a fatal DRAM interface error.	Y	0b	RWC
26	FSB_FE	**Host (FSB) Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal FSB error. 1 = The IMCH detected a fatal FSB error.	Y	0b	RWC
25	NSI_FE	NSI **Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal NSI error. 1 = The IMCH detected a fatal NSI error.	Y	0b	RWC
24	DMA_FE	**DMA Controller Fatal Error Device 1 fatal error (EDMA):** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal DMA Controller error. 1 = The IMCH detected a fatal DMA controller error.	Y	0b	
23	PA_FE	**PCI Express* Port A(0) Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal PCI Express Port A error. 1 = The IMCH detected a fatal PCI Express Port A(0) error.	Y	0b	RWC
22	PA1_FE	**PCI Express Port A1 Fatal Error** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No fatal PCI Express Port A1 error. 1 = The IMCH detected a fatal PCI Express Port A1 error.	Y	0b	RWC
21 : 15	Reserved	Reserved		0b	
14	BUFF_NFE	**Buffer unit detected non-fatal error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal Buffer error. 1 = The IMCH detected a non-fatal Buffer error.	Y	0b	RWC
13	DRAM_NFE	**DRAM Controller Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal DRAM Controller error. 1 = The IMCH detected a non-fatal DRAM Controller error.	Y	0b	RWC
12	FSB_NFE	**FSB Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal FSB error. 1 = The IMCH detected a non-fatal FSB error.	Y	0b	RWC

Table 16-67. Offset 44h: GLOBAL_NERR - Global Next Error Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 44h **Offset End:** 47h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11	NSI_NFE	NSI **Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal NSI error. 1 = The IMCH detected a non-fatal NSI error.		Y	0b	RWC
10	DMA_NFE	**DMA Controller Non-Fatal Error Device 1 non-fatal error (EDMA):** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal DMA Controller error. 1 = The IMCH detected a non-fatal DMA Controller error.		Y	0b	
09	PA_NFE	**PCI Express Port A(0) Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal PCI Express Port A error. 1 = The IMCH detected a non-fatal PCI Express Port A error.		Y	0b	RWC
08	PA1_NFE	**PCI Express Port A1 Non-Fatal Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No non-fatal PCI Express Port A1 error. 1 = The IMCH detected a non-fatal PCI Express Port A1 error.		Y	0b	RWC
07 : 00	Reserved	Reserved			0b	

16.2.1.14 Offset 48h: NSI_FERR - NSI First Error Register

NSI errors for NSI port to IICH. These errors include errors detected on the NSI link, errors from the NSI hierarchy, and errors internal to the NSI unit.

Table 16-68. Offset 48h: NSI_FERR - NSI First Error Register (Sheet 1 of 3)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 48h **Offset End:** 4Bh	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved			0b	
29	UR	**Unsupported Request:** This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unsupported Request detected.			0b	RWC
28	Reserved	Reserved			0b	

Table 16-68. Offset 48h: NSI_FERR - NSI First Error Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 48h **Offset End:** 4Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
27	MTLP	**Malformed TLP Status:** Malformed TLP errors include: data payload length issues, byte enable rule violations, and various other illegal field settings. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Malformed TLP detected.		0b	RWC
26	ROVF	**Receiver Overflow Status:** IMCH checks for overflows on the following upstream queues: posted, non-posted, and completion. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Overflow detected.		0b	RWC
25	UEC	**Unexpected Completion Status:** This bit is set when the device receives a completion which does not correspond to any of the outstanding requests issued by that device. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unexpected Completion detected.		0b	RWC
24	CA	**Completer Abort Status:** If a request received violates the specific programming model of this device, but is otherwise legal, this bit is set. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completer Abort detected.		0b	RWC
23	CT	**Completion Timeout Status:** The Completion Timeout timer must expire if a Request is not completed in 50 ms, but must not expire earlier than 50 µs. When the timer expires, this bit is set. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completion timeout detected.	Y	0b	RWC
22	Reserved	**Reserved**		0b	
21	PTLP	**Poisoned TLP Status:** This bit when set indicates that some portion of the TLP data payload was corrupt. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Poisoned TLP detected.		0b	RWC
20	Reserved	Reserved		0b	
19	DLPE	**Data Link Protocol Error Status:** This bit is set when an ACK/NAK received does not specify the sequence number of an unacknowledged TLP, or of the most recently acknowledged TLP. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Data Link Protocol Error detected.		0b	RWC
18 : 16	Reserved	Reserved		0b	
15	RTTO	**Replay Timer Timeout Status:** The replay timer counts time since the last ACK or NAK DLLP was received. When the timer expires, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Replay Timer timeout detected.	Y	0b	RWC
14	Reserved	Reserved		0b	

Table 16-68. Offset 48h: NSI_FERR - NSI First Error Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 48h **Offset End:** 4Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13	RNRO	**REPLAY_NUM Rollover Status:** A 2-bit counter counts the number of times the retry buffer has been retransmitted. When this counter rolls over, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = REPLAY_NUM rollover detected.	Y	0b	RWC
12	BDLLP	**Bad DLLP Status:** This bit is set when the calculated DLLP CRC is not equal to the received value. Also included are 8b/10b errors within the TLP including wrong disparity. An invalid sequence number also sets this bit. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Bad DLLP detected.	Y	0b	RWC
11	BTLP	**Bad TLP Status:** 0 = The calculated TLP CRC is equal to the received value. 1 = The calculated TLP CRC is not equal to the received value. Also included are 8b/10b errors within the TLP including wrong disparity, and invalid sequence numbers.	Y	0b	RWC
10	Reserved	Reserved		0b	
09	RCVRE	**Receiver Error Status:** Data is delivered over PCI Express via packets built out of 8b/10b symbols. This error is set for problems with the packet framing around these symbols or with symbols received outside of recognized packets. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Error detected.	Y	0b	RWC
08	Reserved	Reserved		0b	
07	FEMR	**Fatal Error Message Received:** 0 = No Fatal Error Message Received over the NSI link. 1 = Fatal Error Message Received over the NSI link.		0b	RWC
06	NEMR	**Non-Fatal Error Message Received:** Non-Fatal Error Message Received over the NSI link. 0 = No Non-Fatal Error Message Received over the NSI link. 1 = Non-Fatal Error Message Received over the NSI link.		0b	RWC
05	CEMR	**Correctable Error Message Received:** Correctable Error Message Received over the NSI link. 0 = No Correctable Error Message Received over the NSI link. 1 = Correctable Error Message Received over the NSI link.		0b	RWC
04 : 03	Reserved	Reserved		0b	
02	PED	**Parity Error Detected during parity conversion from CTB:** Parity Error detected on data received from the core. 0 = No Parity Error detected on data received from the core. 1 = Parity Error detected on data received from the core.		0b	RWC
01	Reserved	Reserved		0b	
00	LD	**Link Down:** 0 = Link has not transitioned from DL_UP to DL_DOWN. 1 = Link transitioned from DL_UP to DL_DOWN.		0b	RWC

16.2.1.15 Offset 4Ch: NSI_NERR - NSI Next Error Register

Errors that are detected after the first error are captured by this register.

Table 16-69. Offset 4Ch: NSI_NERR - NSI Next Error Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 4Ch **Offset End:** 4Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved		0b	
29	UR	**Unsupported Request:** This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unsupported Request detected.		0b	RWC
28	Reserved	Reserved		0b	
27	MTLP	**Malformed TLP Status:** Malformed TLP errors include: data payload length issues, byte enable rule violations, and various other illegal field settings. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Malformed TLP detected.		0b	RWC
26	ROVF	**Receiver Overflow Status:** IMCH checks for overflows on the following upstream queues: posted, non-posted, and completion. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Overflow detected.		0b	RWC
25	UEC	**Unexpected Completion Status:** This bit is set when the device receives a completion which does not correspond to any of the outstanding requests issued by that device. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unexpected Completion detected.		0b	RWC
24	CA	**Completer Abort Status:** If a request received violates the specific programming model of this device, but is otherwise legal, this bit is set. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completer Abort detected.		0b	RWC
23	CT	**Completion Timeout Status:** The Completion Timeout timer must expire if a Request is not completed in 50 ms, but must not expire earlier than 50 µs. When the timer expires, this bit is set. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completion timeout detected.		0b	RWC
22	Reserved	**Reserved**		0b	
21	PTLP	**Poisoned TLP Status:** This bit when set indicates that some portion of the TLP data payload was corrupt. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Poisoned TLP detected.		0b	RWC
20	Reserved	Reserved		0b	
19	DLPE	**Data Link Protocol Error Status:** This bit is set when an ACK/NAK received does not specify the sequence number of an unacknowledged TLP, or of the most recently acknowledged TLP. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Data Link Protocol Error detected.		0b	RWC

Table 16-69. Offset 4Ch: NSI_NERR - NSI Next Error Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 4Ch **Offset End:** 4Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
18 : 16	Reserved	Reserved		0b	
15	RTTO	**Replay Timer Timeout Status:** The replay timer counts time since the last ACK or NAK DLLP was received. When the timer expires, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Replay Timer timeout detected.	Y	0b	RWC
14	Reserved	Reserved		0b	
13	RNRO	**REPLAY_NUM Rollover Status:** A 2-bit counter counts the number of times the retry buffer has been retransmitted. When this counter rolls over, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = REPLAY_NUM rollover detected.	Y	0b	RWC
12	BDLLP	**Bad DLLP Status:** This bit is set when the calculated DLLP CRC is not equal to the received value. Also included are 8b/10b errors within the TLP including wrong disparity. An invalid sequence number also sets this bit. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1= Bad DLLP detected.	Y	0b	RWC
11	BTLP	**Bad TLP Status:** 0 = The calculated TLP CRC is equal to the received value. 1 = The calculated TLP CRC is not equal to the received value. Also included are 8b/10b errors within the TLP including wrong disparity, and invalid sequence numbers.	Y	0b	RWC
10	Reserved	Reserved		0b	
09	RCVRE	**Receiver Error Status:** Data is delivered over PCI Express via packets built out of 8b/10b symbols. This error is set for problems with the packet framing around these symbols or with symbols received outside of recognized packets. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Error detected.	Y	0b	RWC
08	Reserved	Reserved		0b	
07	FEMR	**Fatal Error Message Received:** 0 = No Fatal Error Message Received over the NSI link. 1 = Fatal Error Message Received over the NSI link.		0b	RWC
06	NEMR	**Non-Fatal Error Message Received:** Non-Fatal Error Message Received over the NSI link. 0 = No Non-Fatal Error Message Received over the NSI link. 1 = Non-Fatal Error Message Received over the NSI link.		0b	RWC
05	CEMR	**Correctable Error Message Received:** Correctable Error Message Received over the NSI link. 0 = No Correctable Error Message Received over the NSI link. 1 = Correctable Error Message Received over the NSI link.		0b	RWC
04 : 03	Reserved	Reserved		0b	

Table 16-69. Offset 4Ch: NSI_NERR - NSI Next Error Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 4Ch **Offset End:** 4Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	PED	**Parity Error Detected during parity conversion from CTB:** Parity Error detected on data received from the core. 0 = No Parity Error detected on data received from the core. 1 = Parity Error detected on data received from the core.		0b	RWC
01	Reserved	Reserved		0b	
00	LD	**Link Down:** 0 = Link has not transitioned from DL_UP to DL_DOWN. 1 = Link transitioned from DL_UP to DL_DOWN.		0b	RWC

16.2.1.16 Offset 50h: NSI_SCICMD - NSI SCI Command Register

This register enables various errors to generate an SCI special cycle. When an error flag is set in the FERR or NERR registers, it can generate an SCI special cycle when enabled in the SCICMD registers. Note that one and only one message type can be enabled.

Table 16-70. Offset 50h: NSI_SCICMD - NSI SCI Command Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 50h **Offset End:** 53h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved		0b	
29	UR_SCI	**Generate SCI for** NSI **Error 29:** Generate SCI whenever bit 29 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
28	Reserved	Reserved		0b	
27	MTLP_SCI	**Generate SCI for** NSI **Error 27:** Generate SCI whenever bit 27 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
26	ROVF_SCI	**Generate SCI for** NSI **Error 26:** Generate SCI whenever bit 26 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
25	UEC_SCI	**Generate SCI for** NSI **Error 25:** Generate SCI whenever bit 25 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

Table 16-70. Offset 50h: NSI_SCICMD - NSI SCI Command Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 50h **Offset End:** 53h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
24	CA_SCI	**Generate SCI for** NSI **Error 24:** Generate SCI whenever bit 24 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
23	CT_SCI	**Generate SCI for** NSI **Error 23:** Generate SCI whenever bit 23 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
22	Reserved	Reserved		0b	
21	PTLP_SCI	**Generate SCI for** NSI **Error 21:** Generate SCI whenever bit 21 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
20	Reserved	Reserved		0b	
19	DLPE_SCI	**Generate SCI for** NSI **Error 19:** Generate SCI whenever bit 19 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
18 : 16	Reserved	Reserved		0b	
15	RTTO_SCI	**Generate SCI for** NSI **Error 15:** Generate SCI whenever bit 15 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable	Y	0b	RW
14	Reserved	Reserved		0b	
13	RNRO_SCI	**Generate SCI for** NSI **Error 13:** Generate SCI whenever bit 13 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable	Y	0b	RW
12	BDLLP_SCI	**Generate SCI for** NSI **Error 12:** Generate SCI whenever bit 12 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable	Y	0b	RW
11	BTLP_SCI	**Generate SCI for** NSI **Error 11:** Generate SCI whenever bit 11 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable	Y	0b	RW
10	Reserved	Reserved		0b	
09	RCVRE_SCI	**Generate SCI for** NSI **Error 9:** Generate SCI whenever bit 9 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable	Y	0b	RW
08	Reserved	Reserved		0b	
07	FEMR_SCI	**Generate SCI for** NSI **Error 7:** Generate SCI whenever bit 7 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

Table 16-70. Offset 50h: NSI_SCICMD - NSI SCI Command Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 50h **Offset End:** 53h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	NEMR_SCI	**Generate SCI for** NSI **Error 6:** Generate SCI whenever bit 6 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
05	CEMR_SCI	**Generate SCI for** NSI **Error 5:** Generate SCI whenever bit 5 of NSI _FERR or NSI _NERR is set 0 = Disable 1 = Enable		0b	RW
04 : 03	Reserved	Reserved		0b	
02	PED_SCI	**Generate SCI for** NSI **Error 2:** Generate SCI whenever bit 2 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
01	Reserved	Reserved		0b	
00	LD_SCI	**Generate SCI for** NSI **Error 0:** Generate SCI whenever bit 0 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.17 Offset 54h: NSI_SMICMD - NSI SMI Command Register

This register enables various errors to generate an SMI NSI special cycle. When an error flag is set in the FERR or NERR registers it generates an SMI NSI special cycle when enabled in the SMICMD register. Note that one and only one message type can be enabled.

Table 16-71. Offset 54h: NSI_SMICMD: NSI SMI Command Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 54h **Offset End:** 57h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved		0b	
29	UR_SMI	**Generate SMI for** NSI **Error 29:** Generate SMI whenever bit 29 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
28	Reserved	Reserved		0b	
27	MTLP_SMI	**Generate SMI for** NSI **Error 27:** Generate SMI whenever bit 27 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

Table 16-71. Offset 54h: NSI_SMICMD: NSI SMI Command Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 54h **Offset End:** 57h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
26	ROVF_SMI	**Generate SMI for** NSI **Error 26:** Generate SMI whenever bit 26 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
25	UEC_SMI	**Generate SMI for** NSI **Error 25:** Generate SMI whenever bit 25 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
24	CA_SMI	**Generate SMI for** NSI **Error 24:** Generate SMI whenever bit 24 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
23	CT_SMI	**Generate SMI for** NSI **Error 23:** Generate SMI whenever bit 23 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
22	Reserved	Reserved		0b	
21	PTLP_SMI	**Generate SMI for** NSI **Error 21:** Generate SMI whenever bit 21 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
20	Reserved	Reserved		0b	
19	DLPE_SMI	**Generate SMI for** NSI **Error 19:** Generate SMI whenever bit 19 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
18 : 16	Reserved	Reserved		0b	
15	RTTO_SMI	**Generate SMI for** NSI **Error 15:** Generate SMI whenever bit 15 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
14	Reserved	Reserved		0b	
13	RNRO_SMI	**Generate SMI for** NSI **Error 13:** Generate SMI whenever bit 13 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
12	BDLLP_SMI	**Generate SMI for** NSI **Error 12:** Generate SMI whenever bit 12 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
11	BTLP_SMI	**Generate SMI for** NSI **Error 11:** Generate SMI whenever bit 11 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
10	Reserved	Reserved		0b	
09	RCVRE_SMI	**Generate SMI for** NSI **Error 9:** Generate SMI whenever bit 9 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

Table 16-71. Offset 54h: NSI_SMICMD: NSI SMI Command Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 54h **Offset End:** 57h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	Reserved	Reserved		0b	
07	FEMR_SMI	**Generate SMI for** NSI **Error 7:** Generate SMI whenever bit 6 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
06	NEMR_SMI	**Generate SMI for** NSI **Error 6:** Generate SMI whenever bit 6 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
05	CEMR_SMI	**Generate SMI for** NSI **Error 5:** Generate SMI whenever bit 5 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
04 : 03	Reserved	Reserved		0b	
02	PED_SMI	**Generate SMI for** NSI **Error 2:** Generate SMI whenever bit 2 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
01	Reserved	Reserved		0b	
00	LD_SMI	**Generate SMI for** NSI **Error 0:** Generate SMI whenever bit 0 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.18 Offset 58h: NSI_SERRCMD - NSI SERR Command Register

This register enables various errors to generate an SERR NSI special cycle. When an error flag is set in the FERR or NERR registers it generates an SERR NSI special cycle when enabled in the SERRCMD register. Note that one and only one message type can be enabled.

Table 16-72. Offset 58h: NSI_SERRCMD - NSI SERR Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 58h **Offset End:** 5Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved		0b	
29	UR_SERR	**Generate SERR for** NSI **Error 29:** Generate SERR whenever bit 29 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
28	Reserved	Reserved		0b	
27	MTLP_SERR	**Generate SERR for** NSI **Error 27:** Generate SERR whenever bit 27 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
26	ROVF_SERR	**Generate SERR for** NSI **Error 26:** Generate SERR whenever bit 26 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
25	UEC_SERR	**Generate SERR for** NSI **Error 25:** Generate SERR whenever bit 25 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
24	CA_SERR	**Generate SERR for** NSI **Error 24:** Generate SERR whenever bit 24 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
23	CT_SERR	**Generate SERR for** NSI **Error 23:** Generate SERR whenever bit 23 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
22	Reserved	Reserved		0b	
21	PTLP_SERR	**Generate SERR for** NSI **Error 21:** Generate SERR whenever bit 21 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
20	Reserved	Reserved		0b	
19	DLPE_SERR	**Generate SERR for** NSI **Error 19:** Generate SERR whenever bit 19 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
18 : 16	Reserved	Reserved		0b	
15	RTTO_SERR	**Generate SERR for** NSI **Error 15:** Generate SERR whenever bit 15 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

Table 16-72. Offset 58h: NSI_SERRCMD - NSI SERR Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 58h **Offset End:** 5Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14	Reserved	Reserved		0b	
13	RNRO_SERR	**Generate SERR for** NSI **Error 13:** Generate SERR whenever bit 13 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
12	BDLLP_SERR	**Generate SERR for** NSI **Error 12:** Generate SERR whenever bit 12 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
11	BTLP_SERR	**Generate SERR for** NSI **Error 11:** Generate SERR whenever bit 11 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
10	Reserved	Reserved		0b	
09	RCVRE_SERR	**Generate SERR for** NSI **Error 9:** Generate SERR whenever bit 9 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
08	Reserved	Reserved		0b	
07	FEMR_SERR	**Generate SERR for** NSI **Error 7:** Generate SERR whenever bit 7 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
06	NEMR_SERR	**Generate SERR for** NSI **Error 6:** Generate SERR whenever bit 6 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
05	CEMR_SERR	**Generate SERR for** NSI **Error 5:** Generate SERR whenever bit 5 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
04 : 03	Reserved	Reserved		0b	
02	PED_SERR	**Generate SERR for** NSI **Error 2:** Generate SERR whenever bit 2 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
01	Reserved	Reserved		0b	
00	LD_SERR	**Generate SERR for** NSI **Error 0:** Generate SERR whenever bit 0 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.19 Offset 5Ch: NSI_MCERRCMD - NSI MCERR Command Register

This register enables various errors to generate the MCERR signal on the FSB. When an error flag is set in the FERR or NERR registers it generates a MCERR when enabled in the MCERRCMD.

Table 16-73. Offset 5Ch: NSI_MCERRCMD - NSI MCERR Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 5Ch **Offset End:** 5Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Reserved	Reserved		0b	
29	UR_MCERR	**Generate MCERR for** NSI **Error 29:** Generate MCERR whenever bit 29 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
28	Reserved	Reserved		0b	
27	MTLP_MCERR	**Generate MCERR for** NSI **Error 27:** Generate MCERR whenever bit 27 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
26	ROVF_MCERR	**Generate MCERR for** NSI **Error 26:** Generate MCERR whenever bit 26 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
25	UEC_MCERR	**Generate MCERR for** NSI **Error 25:** Generate MCERR whenever bit 25 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
24	CA_MCERR	**Generate MCERR for** NSI **Error 24:** Generate MCERR whenever bit 24 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
23	CT_MCERR	**Generate MCERR for** NSI **Error 23:** Generate MCERR whenever bit 23 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
22	Reserved	Reserved		0b	
21	PTLP_MCERR	**Generate MCERR for** NSI **Error 21:** Generate MCERR whenever bit 21 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
20	Reserved	Reserved		0b	
19	DLPE_MCERR	**Generate MCERR for** NSI **Error 19:** Generate MCERR whenever bit 19 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
18 : 16	Reserved	Reserved		0b	
15	RTTO_MCERR	**Generate MCERR for** NSI **Error 15:** Generate MCERR whenever bit 15 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
14	Reserved	Reserved		0b	

Table 16-73. Offset 5Ch: NSI_MCERRCMD - NSI MCERR Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 5Ch **Offset End:** 5Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13	RNRO_MCERR	**Generate MCERR for** NSI **Error 13:** Generate MCERR whenever bit 13 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
12	BDLLP_MCERR	**Generate MCERR for** NSI **Error 12:** Generate MCERR whenever bit 12 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
11	BTLP_MCERR	**Generate MCERR for** NSI **Error 11:** Generate MCERR whenever bit 11 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
10	Reserved	Reserved		0b	
09	RCVRE_MCERR	**Generate MCERR for** NSI **Error 9:** Generate MCERR whenever bit 9 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
08	Reserved	Reserved		0b	
07	FEMR_MCERR	**Generate MCERR for** NSI **Error 7:** Generate MCERR whenever bit 7 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
06	NEMR_MCERR	**Generate MCERR for** NSI **Error 6:** Generate MCERR whenever bit 6 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
05	CEMR_MCERR	**Generate MCERR for** NSI **Error 5:** Generate MCERR whenever bit 5 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
04 : 03	Reserved	Reserved		0b	
02	PED_MCERR	**Generate MCERR for** NSI **Error 2:** Generate MCERR whenever bit 2 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW
01	Reserved	Reserved		0b	
00	LD_MCERR	**Generate MCERR for** NSI **Error 0:** Generate MCERR whenever bit 0 of NSI _FERR or NSI _NERR is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.20 Offset 60h: FSB_FERR - FSB First Error Register

This register stores the first error related to the FSB. Only one error bit is set in this register. Any future errors (NEXT errors) are set in the FSB_NERR register. No further error bits in the FSB_FERR register are set until the existing error bit is cleared. These bits are sticky through reset. Software clears these bits by writing a 1 to the bit location.

Note: If multiple errors are reported in the same clock as the first error, all errors are latched.

Table 16-74. Offset 60h: FSB_FERR - FSB First Error Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 60h **Offset End:** 61h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 : 06	Reserved	Reserved		0000b	RWC
05	NDLOCK	**Non-DRAM Lock Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No DRAM Lock Error detected 1 = IMCH detected a lock operation to memory space that did not map into DRAM. (NON-FATAL)	Y	0b	RWC
04	ATOM	**FSB Address Above TOM/TOLM:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB address above TOM/TOLM detected 1 = IMCH has detected an address above the Top of Memory and above 4 Gbyte.If the system has less than 4 Gbyte of DRAM, then unclaimed addresses between TOLM and 4 Gbyte are sent to NSI. (NON-FATAL)	Y	0b	RWC
03	Reserved	Reserved	Y	0b	RWC
02	FSBAGL	**FSB Address Strobe Glitch Detected:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB address strobe glitch detected. 1 = IMCH has detected a glitch one of the FSB address strobes. (FATAL)	Y	0b	RWC
01	FSBDGL	**FSB Data Strobe Glitch Detected:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB data strobe glitch detected. 1 = IMCH has detected a glitch one of the FSB data strobes. (FATAL)	Y	0b	RWC
00	Reserved	Reserved	Y	0b	RWC

16.2.1.21 Offset 62h: FSB_NERR - FSB Next Error Register

Table 16-75. Offset 62h: FSB_NERR - FSB Next Error Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 62h **Offset End:** 63h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved		0000b	RWC
05	NDLOCK	**Non-DRAM Lock Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No DRAM Lock Error detected 1 = IMCH detected a lock operation to memory space that did not map into DRAM. (NON-FATAL)	Y	0b	RWC
04	ATOM	**FSB Address Above TOM/TOLM:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB address above TOM/TOLM detected 1 = IMCH has detected an address above the Top of Memory and above 4 Gbyte.If the system has less than 4 Gbyte of DRAM, then unclaimed addresses between TOLM and 4 Gbyte are sent to NSI. (NON-FATAL)	Y	0b	RWC
03	Reserved	Reserved	Y	0b	RWC
02	FSBAGL	**FSB Address Strobe Glitch Detected:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB address strobe glitch detected. 1 = IMCH has detected a glitch one of the FSB address strobes. (FATAL)	Y	0b	RWC
01	FSBDGL	**FSB Data Strobe Glitch Detected:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No FSB data strobe glitch detected. 1 = IMCH has detected a glitch one of the FSB data strobes. (FATAL)	Y	0b	RWC
00	Reserved	Reserved	Y	0b	RWC

16.2.1.22 Offset 64h: FSB_EMASK - FSB Error Mask Register

This register masks the FSB unit errors from being recognized, preventing them from being logged at the unit or global level, and no interrupt/messages are generated. These bits are sticky through reset.

Table 16-76. Offset 64h: FSB_EMASK - FSB Error Mask Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 64h **Offset End:** 65h	
Size: 16 bit	**Default:**	0009h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved	Y	0000b	RW
05	NDLOCKM	**Non-DRAM Lock Error Mask:** This bit is sticky through reset. 0 = Enable Non-DRAM Lock Error detection and reporting 1 = Mask Non-DRAM Lock Error detection and reporting	Y	0b	RW
04	ATOMM	**FSB Address Above TOM Mask:** This bit is sticky through reset. 0 = Enable FSB address above TOM detection and reporting 1 = Mask address above TOM detection and reporting	Y	0b	RW
03	Reserved	Reserved	Y	1b	RW
02	FSBAGLM	**FSB Address Strobe Glitch Detected Mask:** This bit is sticky through reset. 0 = Enable FSB address strobe glitch detection and reporting 1 = Mask address strobe glitch detection and reporting	Y	0b	RW
01	FSBDGLM	**FSB Data Strobe Glitch Detected Mask:** This bit is sticky through reset. 0 = Enable FSB data strobe glitch detection and reporting 1 = Mask data strobe glitch detection and reporting	Y	0b	RW
00	Reserved	Reserved	Y	1b	RW

16.2.1.23 Offset 68h: FSB_SCICMD - FSB SCI Command Register

This register enables various errors to generate an SCI NSI special cycle. When an error flag is set in the FERR or NERR registers, it generates an SCI NSI special cycle when enabled in the SCICMD register. Note that one and only one message type can be enabled.

Table 16-77. offset 68h: FSB_SCICMD - FSB SCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 68h **Offset End:** 69h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved		0000b	RW
05	NDLOCK_SCI	**Non-DRAM Lock Error SCI Enable:** Controls whether or not an SCI is generated when bit 5 of the FSB_FERR or FSB_NERR register is set. 0 = No SCI generated on Non-DRAM Lock Error detection 1 = Enable SCI generation on Non-DRAM Lock Error detection		0b	RW
04	ATOM_SCI	**FSB Address Above TOM SCI Enable:** Controls whether or not an SCI is generated when bit 4 of the FSB_FERR or FSB_NERR register is set. 0 = No SCI generated on FSB address above TOM detection 1 = Enable SCI generation on FSB address above TOM detection		0b	RW
03	Reserved	Reserved		0b	RW
02	FSBAGL_SCI	**FSB Address Strobe Glitch Detected SCI Enable:** Controls whether or not an SCI is generated when bit 2 of the FSB_FERR or FSB_NERR register is set. 0 = No SCI generated on FSB address strobe glitch detection 1 = Enable SCI generation on FSB address strobe glitch detection		0b	RW
01	FSBDGL_SCI	**FSB Data Strobe Glitch Detected SCI Enable:** Controls whether or not an SCI is generated when bit 1 of the FSB_FERR or FSB_NERR register is set. 0 = No SCI generated on FSB data strobe glitch detection 1 = Enable SCI generation on FSB data strobe glitch detection		0b	RW
00	Reserved	Reserved		0b	RW

16.2.1.24 Offset 6Ah: FSB_SMICMD - FSB SMI Command Register

This register enables various errors to generate an SMI NSI special cycle. When an error flag is set in the FSB_FERR or FSB_NERR register, it generates an SMI NSI special cycle when enabled in the SMICMD register. Note that one and only one message type can be enabled.

Table 16-78. Offset 6Ah: FSB_SMICMD - FSB SMI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 6Ah **Offset End:** 6Bh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved		0000b	RW
05	NDLOCK_SMI	**Non-DRAM Lock Error SMI Enable:** Controls whether or not an SMI is generated when bit 5 of the FSB_FERR or FSB_NERR register is set. 0 = No SMI generated on Non-DRAM Lock Error detection 1 = Enable SMI generation on Non-DRAM Lock Error detection		0b	RW
04	ATOM_SMI	**FSB Address Above TOM SMI Enable:** Controls whether or not an SMI is generated when bit 4 of the FSB_FERR or FSB_NERR register is set. 0 = No SMI generated on FSB address above TOM detection 1 = Enable SMI generation on FSB address above TOM detection		0b	RW
03	Reserved	Reserved		0b	RW
02	FSBAGL_SMI	**FSB Address Strobe Glitch Detected SMI Enable:** Controls whether or not an SMI is generated when bit 2 of the FSB_FERR or FSB_NERR register is set. 0 = No SMI generated on FSB address strobe glitch detection 1 = Enable SMI generation on FSB address strobe glitch detection		0b	RW
01	FSBDGL_SMI	**FSB Data Strobe Glitch Detected SMI Enable:** Controls whether or not an SMI is generated when bit 1 of the FSB_FERR or FSB_NERR register is set. 0 = No SMI generated on FSB data strobe glitch detection 1 = Enable SMI generation on FSB data strobe glitch detection		0b	RW
00	Reserved	Reserved		0b	RW

16.2.1.25 Offset 6Ch: FSB_SERRCMD - FSB SERR Command Register

This register enables various errors to generate an SERR NSI special cycle. When an error flag is set in the FSB_FERR or FSB_NERR register, it generates an SERR NSI special cycle when enabled in the SERRCMD register. Note that one and only one message type can be enabled.

Table 16-79. Offset 6Ch: FSB_SERRCMD - FSB SERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 6Ch **Offset End:** 6Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved		0000b	RW
05	NDLOCK_SERR	**Non-DRAM Lock Error SERR Enable:** Controls whether or not an SERR is generated when bit 5 of the FSB_FERR or FSB_NERR register is set. 0 = No SERR generated on Non-DRAM Lock Error detection 1 = Enable SERR generation on Non-DRAM Lock Error detection		0b	RW
04	ATOM_SERR	**FSB Address Above TOM SERR Enable:** Controls whether or not an SERR is generated when bit 4 of the FSB_FERR or FSB_NERR register is set. 0 = No SERR generated on FSB address above TOM detection 1 = Enable SERR generation on FSB address above TOM detection		0b	RW
03	Reserved	Reserved		0b	RW
02	FSBAGL_SERR	**FSB Address Strobe Glitch Detected SERR Enable:** Controls whether or not an SERR is generated when bit 2 of the FSB_FERR or FSB_NERR register is set. 0 = No SERR generated on FSB address strobe glitch detection 1 = Enable SERR generation on FSB address strobe glitch detection		0b	RW
01	FSBDGL_SERR	**FSB Data Strobe Glitch Detected SERR Enable:** Controls whether or not an SERR is generated when bit 1 of the FSB_FERR or FSB_NERR register is set. 0 = No SERR generated on FSB data strobe glitch detection 1 = Enable SERR generation on FSB data strobe glitch detection		0b	RW
00	Reserved	Reserved		0b	RW

16.2.1.26 Offset 6Eh: FSB_MCERRCMD - FSB MCERR Command Register

This register enables various errors to generate the MCERR signal on the FSB. When an error flag is set in the FSB_FERR or FSB_NERR register, it generates a MCERR# when enabled in the MCERRCMD.

Table 16-80. Offset 6Eh: FSB_MCERRCMD - FSB MCERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 6Eh **Offset End:** 6Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	
09 06	Reserved	Reserved		0000b	RW
05	NDLOCK_MCERR_N	**Non-DRAM Lock Error MCERR# Enable:** Controls whether or not an MCERR# is generated when bit 5 of the FSB_FERR or FSB_NERR register is set. 0 = No MCERR# generated on Non-DRAM Lock Error detection 1 = Enable MCERR# generation on Non-DRAM Lock Error detection		0b	RW
04	ATOM_MCERR_N	**FSB Address Above TOM MCERR# Enable:** Controls whether or not an MCERR# is generated when bit 4 of the FSB_FERR or FSB_NERR register is set. 0 = No MCERR# generated on FSB address above TOM detection 1 = Enable MCERR# generation on FSB address above TOM detection		0b	RW
03	Reserved	Reserved		0b	RW
02	FSBAGL_MCERR_N	**FSB Address Strobe Glitch Detected MCERR# Enable:** Controls whether or not an MCERR# is generated when bit 2 of the FSB_FERR or FSB_NERR register is set. 0 = No MCERR# generated on FSB address strobe glitch detection 1 = Enable MCERR# generation on FSB address strobe glitch detection		0b	RW
01	FSBDGL_MCERR_N	**FSB Data Strobe Glitch Detected MCERR# Enable:** Controls whether or not an MCERR# is generated when bit 1 of the FSB_FERR or FSB_NERR register is set. 0 = No MCERR# generated on FSB data strobe glitch detection 1 = Enable MCERR# generation on FSB data strobe glitch detection		0b	RW
0	Reserved	Reserved		0b	RW

16.2.1.27 Offset 70h: BUF_FERR - Memory Buffer First Error Register

Signals errors occurring in the memory system coherent Posted Memory Write Buffer (PMWB).

Table 16-81. Offset 70h: BUF_FERR - Memory Buffer First Error Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 70h **Offset End:** 70h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03	DPMWB	Internal DRAM to PMWB Parity Error Detected: 0 = Parity error not detected. 1= Parity error detected when a cacheline read from DRAM was written to the PMWB as part of a Read/Modify/Write operation (partial write). (NON-FATAL)		0b	RWC
02	IOPMWB	**Internal System Bus or I/O to PMWB Parity Error Detected:** 0 = Parity error not detected. 1 = Parity error detected on a line write to PMWB. (NON-FATAL)		0b	RWC
01	PMWBSYS	Internal PMWB to System Bus Parity Error Detected: 0 = Parity error not detected. 1 = Parity error detected on data to the System Bus. (NON-FATAL)		0b	RWC
00	PMWBD	**Internal PMWB to DRAM Parity Error Detected:** 0 = Parity error not detected. 1 = Parity error detected when PMWB is flushed to DRAM. (NON-FATAL)		0b	RWC

16.2.1.28 Offset 72h: BUF_NERR - Memory Buffer Next Error Register

Table 16-82. Offset 72h: BUF_NERR - Memory Buffer Next Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 72h **Offset End:** 72h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03	DPMWB	Internal DRAM to PMWB Parity Error Detected: 0 = Parity error not detected. 1 = Parity error detected when a cacheline read from DRAM was written to the PMWB as part of a Read/Modify/ Write operation (partial write). (NON-FATAL)		0b	RWC

Table 16-82. Offset 72h: BUF_NERR - Memory Buffer Next Error Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 72h **Offset End:** 72h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
02	IOPMWB	**Internal System Bus or I/O to PMWB Parity Error Detected:** 0 = Parity error not detected. 1 = Parity error detected on a line write to PMWB. (NON-FATAL)			0b	RWC
01	PMWBSYS	Internal PMWB to System Bus Parity Error Detected: 0 = Parity error not detected. 1 = Parity error detected on data to the System Bus. (NON-FATAL)			0b	RWC
00	PMWBD	**Internal PMWB to DRAM Parity Error Detected:** 0 = Parity error not detected. 1= Parity error detected when PMWB is flushed to DRAM. (NON-FATAL)			0b	RWC

16.2.1.29 Offset 74h: BUF_EMASK - Memory Buffer Error Mask Register

This register masks the unit errors from being recognized. Because they are not recognized, they are not logged at the unit or global level and no interrupt/messages are generated.

Table 16-83. Offset 74h: BUF_EMASK - Memory Buffer Error Mask Register

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1		**Offset Start:** 74h **Offset End:** 74h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved			0b	
03	BUF_EMASK03	Mask Error bit 03: 0 = Disable mask. 1 = Enable mask.			0b	RW
02	BUF_EMASK02	Mask Error bit 02: 0 = Disable mask. 1 = Enable mask.			0b	RW
01	BUF_EMASK01	Mask Error bit 01: 0 = Disable mask. 1 = Enable mask.			0b	RW
00	BUF_EMASK00	Mask Error bit 00: 0 = Disable mask. 1 = Enable mask.			0b	RW

16.2.1.30 Offset 78h: BUF_SCICMD - Memory Buffer SCI Command Register

This register enables various errors to generate an SCI NSI special cycle. When an error flag is set in the FERR or NERR registers, it generates an SCI NSI special cycle when enabled in the SCICMD registers. Note that one and only one message type can be enabled.

Table 16-84. Offset 78h: BUF_SCICMD - Memory Buffer SCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 78h **Offset End:** 78h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03	DPMWB_SCI	**Internal DRAM Interface to PMWB Parity Error SCI Enable:** Generate SCI when parity error detected for DRAM Interface to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
02	IOPMWB_SCI	**Internal System Bus or I/O to PMWB Parity Error SCI Enable:** Generate SCI when parity error detected for internal System Bus or I/O to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
01	PMWBSYS_SCI	**Internal PMWB to System Bus Parity Error SCI Enable:** Generate SCI when parity error detected for PMWB to System Bus when this bit is set. 0 = Disable 1 = Enable		0b	RW
00	PMWBD_SCI	**Internal PMWB to DRAM I/F Parity Error SCI Enable:** Generate SCI when parity error detected for PMWB to DRAM I/F when this bit is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.31 Offset 7Ah: BUF_SMICMD - Memory Buffer SMI Command Register

This register enables various errors to generate an SMI NSI special cycle. When an error flag is set in the FERR or NERR registers, it generates an SMI NSI special cycle when enabled in the SMICMD register. Note that one and only one message type can be enabled.

Table 16-85. Offset 7Ah: BUF_SMICMD - Memory Buffer SMI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 7Ah **Offset End:** 7Ah	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03	DPMWB_SMI	**Internal DRAM Interface to PMWB Parity Error SMI Enable:** Generate SMI when parity error detected for DRAM interface to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
02	IOPMWB_SMI	**Internal System Bus or I/O to PMWB Parity Error SMI Enable:** Generate SMI when parity error detected for internal System Bus or I/O to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
01	PMWBSYS_SMI	**Internal PMWB to System Bus Parity Error SMI Enable:** Generate SMI when parity error detected for PMWB to System Bus when this bit is set. 0 = Disable 1 = Enable		0b	RW
00	PMWBD_SMI	**Internal PMWB to DRAM I/F Parity Error SMI Enable:** Generate SMI when parity error detected for PMWB to DRAM I/F when this bit is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.32 Offset 7Ch: BUF_SERRCMD - Memory Buffer SERR Command Register

This register enables various errors to generate an SERR NSI special cycle. When an error flag is set in the FERR or NERR registers, it generates an SERR NSI special cycle when enabled in the SERRCMD register. Note that one and only one message type can be enabled.

Table 16-86. Offset 7Ch: BUF_SERRCMD - Memory Buffer SERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 7Ch **Offset End:** 7Ch	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03	DPMWB_SERR	**Internal DRAM I/F to PMWB Parity Error SERR Enable:** Generate SERR when parity error detected for DRAM I/F to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
02	IOPMWB_SERR	**Internal System Bus or I/O to PMWB Parity Error SERR Enable:** Generate SERR when parity error detected on write port 0 when this bit is set. 0 = Disable 1 = Enable		0b	RW
01	PMWBSYS_SERR	**Internal PMWB to System Bus Parity Error SERR Enable:** Generate SERR when parity error detected for PMWB to System Bus when this bit is set. 0 = Disable 1 = Enable		0b	RW
00	PMWBD_SERR	**Internal PMWB to DRAM I/F Parity Error SERR Enable:** Generate SERR when parity error detected for PMWB to DRAM I/F when this bit is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.33 Offset 7Eh: BUF_MCERRCMD - Memory Buffer MCERR Command Register

This register enables various errors to generate a MCERR signal on the FSB. When an error flag is set in the FERR or NERR registers, it generates a MCERR when enabled in the MCERRCMD register.

Table 16-87. Offset 7Eh: BUF_MCERRCMD - Memory Buffer MCERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 7Eh **Offset End:** 7Eh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		00h	
03	DPMWB_MCERR	**Internal DRAM I/F to PMWB Parity Error MCERR Enable:** Generate MCERR when parity error detected for DRAM I/F to PMWB when this bit is set. 0 = Disable 1 = Enable		0b	RW
02	IOPMWB_MCERR	**Internal System Bus or I/O to PMWB Parity Error MCERR Enable:** Generate MCERR when parity error detected on write port 0 when this bit is set. 0 = Disable 1 = Enable		0b	RW
01	PMWBSYS_MCERR	**Internal PMWB to System Bus Parity Error MCERR Enable:** Generate MCERR when parity error detected for PMWB to System Bus when this bit is set. 0 = Disable 1 = Enable		0b	RW
00	PMWBD_MCERR	**Internal PMWB to DRAM I/F Parity Error MCERR Enable:** Generate MCERR when parity error detected for PMWB to DRAM I/F when this bit is set. 0 = Disable 1 = Enable		0b	RW

16.2.1.34 Offset E4h: NSIERRINJCTL - NSI Error Injection Control Register

This register controls the way in which the IMCH handles parity errors on the Interface.

Table 16-88. Offset E4h: NSIERRINJCTL - NSI Error Injection Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** E4h **Offset End:** E7h	
Size: 32 bit	**Default:**	00040000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		00h	
19	STPSCRM	**Stop and Scream bit:** This is a special control for errors going to NSI, outgoing from the IMCH core. 0 = Outgoing data errors are propagated. 1 = Outgoing data errors are reported, but not propagated.		0b	RW
18	EnDP	**Enable data poisoning:** This bit controls whether or not the IMCH marks data as “poisoned” when a parity error is detected from the NSI. 0 = Error checking disabled. 1 = Error Checking Enabled. Incoming data with parity errors are marked as “poisoned” before being sent on towards its destination.		1b	RW
17 : 00	Reserved	Reserved		0b	

16.2.1.35 Offset E8h: BERRINJCTL - Buffer Error Injection Control Register

This register enables the injection of errors on data read out of the posted write buffer. The lower 16 bits are the corresponding flip parity bits for the cacheline of data. The upper bits in the register are for the use and control of the associated flip parity bits.

Table 16-89. Offset E8h: BERRINJCTL - Buffer Error Injection Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** E8h **Offset End:** EBh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 19	Reserved	Reserved		00h	
18	EnDP	**Enable / Disable data poisoning:** When = '0' , Data Poisoning is disabled - When a parity error is detected it is sent to memory with good parity. An interrupt will be generated if enabled and not masked. (This is the Plumas/Placer model) NOTE: This creates a race condition between when data could be used vs. reporting and responding to the interrupt. When = '1', Data Poisoning is enabled - When a parity error is detected data is sent to memory with bad parity. An interrupt will be generated if enabled and not masked. (This is the new Cayuse model) NOTE: Error Injection is possible regardless of this bit setting.		0b	RW
17 : 00	Reserved	Reserved		0b	

16.2.1.36 Offset 80h: DRAM_FERR - DRAM First Error Register

This register signals the first error occurring in the memory system. Refer to Section 11.5, "Error Handling" to understand the error handling mechanism implemented in Core.

This register stores the first error related to the DRAM Controller. Typically, only one error bit is set in this register. However, in the case of multiple errors in the same cycle, multiple bits can be set in this register.

No further error bits in the DRAM_FERR register are set until the existing error bit is cleared by software. Any future errors (NEXT errors) are set in the DRAM_NERR register defined in Table 16-91.

The bits defined in this register are sticky through reset. Software clears these bits by writing a 1 to the bit location.

The errors in this register are reported up into the GLOBAL_FERR registers as either "fatal" or "non-fatal" errors from the memory controller as noted in the descriptions below.

Note: All memory controller errors are "not-fatal".

Note: Logging of these errors can be masked only by setting the corresponding bit in Section 16.2.1.38, "Offset 84h: DRAM_EMASK - DRAM Error Mask Register".

Table 16-90. Offset 80h: DRAM_FERR - DRAM First Error Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** 80h **Offset End:** 81h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	Reserved	Reserved		00h	RO
07	MTCA	**Memory Test Complete:** Not an error condition. This bit is set by hardware to signal BIOS that hardware testing of the channel is complete. This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 1 = Hardware-based test of DRAM is complete. (NON-FATAL)	Y	0b	RWC
06	UERRA	**Uncorrectable Error on Write:** (Uncorrectable) This bit is set on a detected error regardless of ECC mode, even if ECC is disabled. However if the error was injected via ECCDIAG, this bit is not set. Note that the state of the ECCDIAG.MEMPEN does not impact the setting of this bit. For more details please see Section 16.1.1.44, "Offset 84h: ECCDIAG – ECC Detection/ Correction Diagnostic Register". This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No parity error detected on writes to DRAM. 1 = Parity error detected on write to DRAM. (NON-FATAL)	Y	0b	RWC
05 :04	Reserved	Reserved		00b	RO
03	ETDA	**Error Threshold Detect:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. This bit can be set by either a SEC or DED event, if the corresponding error counter is set. 0 = No Error Threshold detected 1 = Error Threshold detected. (NON-FATAL)	Y	0b	RWC
02	USDEA	**Uncorrectable Scrubber Data Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Scrubber Error Detected 1 = Scrubber Error Detected. (NON-FATAL)	Y	0b	RWC
01	URMEA	**Uncorrectable Read Memory Error:** (Uncorrectable) Applies to non-scrub demand (normal demand fetch) reads. This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Uncorrectable Non-Scrub Demand Read Memory Error 1 = Uncorrectable Non-Scrub Demand Read Memory Error. (NON-FATAL)	Y	0b	RWC
00	CRMEA	**Correctable read memory Error:** (Correctable) SECs (Single Bit Error Correction) detected by normal demand requests or scrub/demand fetch (normal read to memory). This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Correctable Read Memory Error. 1 = Correctable Read Memory Error. (NON-FATAL)	Y	0b	RWC

16.2.1.37 Offset 82h: DRAM_NERR - DRAM Next Error Register

This register captures the errors related to the DRAM Controller that occur after the first error is detected and captured in DRAM_FERR. Refer to Section 11.5, "Error Handling" to understand the error handling mechanism implemented in the memory controller.

Multiple bits can be set in this register if multiple errors occur following the first error prior to software clearing the first error register. These bits are sticky through reset. Software clears these bits by writing a 1 to the bit location.

The errors in this register are reported up into the GLOBAL_NERR registers as either "fatal" or "non-fatal" errors from the memory controller as noted in the descriptions below.

Unlike the DRAM_FERR register, multiple errors are accumulated in the DRAM_NERR register.

Note: All memory controller errors are "not-fatal".

Note: Logging of these errors can be masked only by setting the corresponding bit in Section 16.2.1.38, "Offset 84h: DRAM_EMASK - DRAM Error Mask Register".

Table 16-91. Offset 82h: DRAM_NERR - DRAM Next Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** 82h **Offset End:** 83h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	Reserved	Reserved		00h	RO
07	MTCA	**Memory Test Complete:** Not an error condition. This bit is set by hardware to signal BIOS that hardware testing of the channel is complete. This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 1 = Hardware-based test of DRAM is complete. (NON-FATAL)	Y	0b	RWC
06	UERRA	**Uncorrectable Error on Write:** (Uncorrectable) This bit is set on a detected error regardless of ECC mode, even if ECC is disabled. However if the error was injected via ECCDIAG, this bit is not set. Note that the state of the ECCDIAG.MEMPEN does not impact the setting of this bit. For more details please see Section 16.1.1.44, "Offset 84h: ECCDIAG – ECC Detection/ Correction Diagnostic Register" This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No parity error detected on writes to DRAM. 1 = Parity error detected on write to DRAM. (NON-FATAL)	Y	0b	RWC
05 :04	Reserved	Reserved		00b	RO

Table 16-91. Offset 82h: DRAM_NERR - DRAM Next Error Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 82h **Offset End:** 83h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	ETDA	**Error Threshold Detect:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. This bit can be set by either a SEC or DED event, if the corresponding error counter is set. The bit can also set if the DRAM_SEC current error count register is cleared, via a software diagnostic error count write. 0 = No Error Threshold detected 1 = Error Threshold detected. (NON-FATAL)	Y	0b	RWC
02	USDEA	**Uncorrectable Scrubber Data Error:** This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Scrubber Error Detected 1 = Scrubber Error Detected. (NON-FATAL)	Y	0b	RWC
01	URMEA	**Uncorrectable Read Memory Error:** (Uncorrectable) Applies to non-scrub demand (normal demand fetch) reads. This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Uncorrectable Non-Scrub Demand Read Memory Error 1 = Uncorrectable Non-Scrub Demand Read Memory Error. (NON-FATAL)	Y	0b	RWC
00	CRMEA	**Correctable read memory Error:** (Correctable) SECs (Single Bit Error Correction) detected by normal demand requests or scrub/demand fetch (normal read to memory). This bit is sticky through reset. System software clears this bit by writing a 1 to the location. 0 = No Correctable Read Memory Error. 1 = Correctable Read Memory Error. (NON-FATAL)	Y	0b	RWC

16.2.1.38 Offset 84h: DRAM_EMASK - DRAM Error Mask Register

This register masks the DRAM Controller errors and events from being recognized, preventing them from being logged at either the unit level (via the DRAM_FERR or DRAM_NERR registers, see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”) or global level (via GLOBAL_FERR or GLOBAL_NERR)and preventing an interrupt/messages from being generated.

These bits are sticky through reset.

Note: If ETD_MASK is changed from 0 to 1 and any error count is already above threshold, then the error(s) will be immediately reported via FERR/NERR.

Table 16-92. Offset 84h: DRAM_EMASK - DRAM Error Mask Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 84h **Offset End:** 84h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MTC_MASK	**Memory Test Complete Mask:** This bit is sticky through reset. 0 = Allow Memory Test Complete logging and signaling. 1 = Mask Memory Test Complete logging and signaling. Mask Error bits 7.	Y	0b	RW
06	UERR_MASK	**Uncorrectable Error Detected on Write to DRAM Mask:** This bit is sticky through reset. 0 = Allow Poisoned Write to DRAM detection and signaling. 1 = Mask Poisoned Write to DRAM detection and signaling. Mask Error bit 6.	Y	0b	RW
05 :04	Reserved	Reserved		00b	RW
03	ETD_MASK	**Error Threshold Detect Mask:** This bit is sticky through reset. 0 = Allow Error Threshold detection and signaling. 1 = Mask Error Threshold detection and signaling. Mask Error bit 3.	Y	0b	RW
02	SDE_MASK	**Scrubber Data Error Mask:** This bit is sticky through reset. 0 = Allow Scrubber Data Error detection and signaling. 1 = Mask Scrubber Data Error detection and signaling. Mask Error bit 2.	Y	0b	RW
01	URME_MASk	**Uncorrectable Read Memory Error Mask:** This bit is sticky through reset. 0 = Allow Uncorrectable Memory Read Error detection and signaling. 1 = Mask Uncorrectable Memory Read Error detection and signaling. Mask Error bit 1.	Y	0b	RW
00	CRME_MASK	**Correctable Read Memory Error Mask:** This bit is sticky through reset. 0 = Allow Correctable Memory Read Error detection and signaling. 1 = Mask Correctable Memory Read Error detection and signaling. Mask Error bit 0.	Y	0b	RW

16.2.1.39 Offset 88h: DRAM_SCICMD - DRAM SCI Command Register

This register enables the memory controller to generate an SCI NSI special cycle for various error flags. When an error flag is set in either the DRAM_FERR or DRAM_NERR registers (see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”), hardware generates an SCI NSI special cycle when enabled in the DRAM_SCICMD register.

Note that software should enable one and only one message type for a given error flag.

Table 16-93. Offset 88h: DRAM_SCICMD - DRAM SCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 88h **Offset End:** 88h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MTC_SCI	**Memory Test Complete SCI Enable:** Generate SCI when Bit 7 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
06	PWD_SCI	**Poisoned Write to DRAM SCI Enable:** Generate SCI when Bit 6 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
05 :04	Reserved	Reserved	N	00b	RO
03	ETD_SCI	**Error Threshold Detect SCI Enable:** Generate SCI when Bit 3 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
02	SDE_SCI	**Scrubber Data Error SCI Enable:** Generate SCI when Bit 2 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
01	URME_SCI	**Uncorrectable Read Memory Error SCI Enable:** Generate SCI when Bit 1 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
00	CRME_SCI	**Correctable Read Memory Error SCI Enable:** Generate SCI when Bit 0 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW

16.2.1.40 Offset 8Ah: DRAM_SMICMD - DRAM SMI Command Register

This register enables the memory controller to generate an SMI NSI special cycle for various error flags. When an error flag is set in either the DRAM_FERR or DRAM_NERR registers (see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”), hardware generates an SMI NSI special cycle when enabled in the DRAM_SMICMD register.

Note that software should enable one and only one message type for a given error flag.

Table 16-94. Offset 8Ah: DRAM_SMICMD - DRAM SMI Command Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** 8Ah **Offset End:** 8Ah	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MTC_SMI	**Memory Test Complete SMI Enable:** Generate SMI when Bit 7 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
06	PWD_SMI	**Poisoned Write to DRAM SMI Enable:** Generate SMI when Bit 6 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
05 :04	Reserved	Reserved	N	00b	RO
03	ETD_SMI	**Error Threshold Detect SMI Enable:** Generate SMI when Bit 3 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
02	SDE_SMI	**Scrubber Data Error SMI Enable:** Generate SMI when Bit 2 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
01	URME_SMI	**Uncorrectable Read Memory Error SMI Enable:** Generate SMI when Bit 1 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
00	CRME_SMI	**Correctable Read Memory Error SMI Enable:** Generate SMI when Bit 0 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW

16.2.1.41 Offset 8Ch: DRAM_SERRCMD - DRAM SERR Command Register

This register enables the memory controller to generate an SERR NSI special cycle for various error flags. When an error flag is set in either the DRAM_FERR or DRAM_NERR registers (see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”), hardware generates an SERR NSI special cycle when enabled in the DRAM_SERRCMD register.

Note: Software should enable one and only one message type for a given error flag.

Table 16-95. Offset 8Ch: DRAM_SERRCMD - DRAM SERR Command Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** 8Ch **Offset End:** 8Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MTC_SERR	**Memory Test Complete SERR Enable:** Generate SERR when Bit7 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
06	PWD_SERR	**Poisoned Write to DRAM SERR Enable:** Generate SERR when Bit 6 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
05 :04	Reserved	Reserved	N	00b	RO
03	ETD_SERR	**Error Threshold Detect SERR Enable:** Generate SERR when Bit 3 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
02	SDE_SERR	**Scrubber Data Error SERR Enable:** Generate SERR when Bit2 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
01	URME_SERR	**Uncorrectable Read Memory Error SERR Enable:** Generate SERR when Bit 1 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
00	CRME_SERR	**Correctable Read Memory Error SERR Enable:** Generate SERR when Bit 0 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW

16.2.1.42 Offset 8Eh: DRAM_MCERRCMD - DRAM MCERR Command Register

This register enables the memory controller to generate an MCERR# signal on the FSB for various error flags. When an error flag is set in either the DRAM_FERR or DRAM_NERR registers (see Section 16.2.1.36, "Offset 80h: DRAM_FERR - DRAM First Error Register" and Section 16.2.1.37, "Offset 82h: DRAM_NERR - DRAM Next Error Register"), hardware generates an MCERR# signal on the FSB when enabled in the DRAM_MCERRCMD register.

Note that software should enable one and only one message type for a given error flag.

Table 16-96. Offset 8Eh: DRAM_MCERRCMD - DRAM MCERR Command Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** 8Eh **Offset End:** 8Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MTC_MCERR	**Memory Test Complete MCERR# Enable:** Generate MCERR# when Bit 7 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
06	PWD_MCERR	**Poisoned Write to DRAM MCERR# Enable:** Generate MCERR# when Bit 6 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
05 :04	Reserved	Reserved	N	00b	RO
03	ETD_MCERR	**Error Threshold Detect MCERR# Enable:** Generate MCERR# when Bit 3 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
02	SDE_MCERR	**Scrubber Data Error MCERR# Enable:** Generate MCERR# when Bit 2 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
01	URM_MCERR	**Uncorrectable Read Memory ErrorMCERR# Enable:** Generate MCERR# when Bit 1 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW
00	CRM_MCERR	**Correctable Read Memory Error MCERR# Enable:** Generate MCERR# when Bit 0 of DRAM_FERR or DRAM_NERR is set. 0 = Disable 1 = Enable	N	0b	RW

16.2.1.43 Offset 98h: THRESH_SEC0 - Rank 0 SEC Error Threshold Register

Threshold compare value for SEC errors. An Error Threshold Detect event is signaled via bit 3 of DRAM_FERR or DRAM_NERR (see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”) if the rank 0 SEC counter (see Section 16.2.1.48, “Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register”) exceeds the value programmed into this register. The bits in this register are sticky through reset.

Table 16-97. Offset 98h: THRESH_SEC0 - Rank 0 SEC Error Threshold Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 98h **Offset End:** 99h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	THRSH_SEC0	**Threshold:** Threshold compare value for rank 0 SEC errors.	Y	0000h	RW

16.2.1.44 Offset 9Ah: THRESH_SEC1 - Rank 1 SEC Error Threshold Register

Threshold compare value for SEC errors. An Error Threshold Detect event is signaled via bit 3 of DRAM_FERR or DRAM_NERR (see Section 16.2.1.36, “Offset 80h: DRAM_FERR - DRAM First Error Register” and Section 16.2.1.37, “Offset 82h: DRAM_NERR - DRAM Next Error Register”) if the rank 1 SEC counter (see Section 16.2.1.50, “Offset B4h: DRAM_SEC_R1 - DRAM Rank 1 SEC Error Counter Register”) exceeds the value programmed into this register. The bits in this register are sticky through reset.

Table 16-98. Offset 9Ah: THRESH_SEC1 - Rank 1 SEC Error Threshold Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** 9Ah **Offset End:** 9Bh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	THRSH_SEC1	**Threshold:** Threshold compare value for rank 1 SEC errors.	Y	0000h	RW

16.2.1.45 Offset A0h: DRAM_SECF_ADD - DRAM First Single Bit Error Correct Address Register

Captures the address of the SEC error occurring in the memory system (including scrubs). The value in this register is only valid if the Correctable Read Memory Error bit in the DRAM_FERR register has been set. The bits in this register are sticky through reset. (see Section 16.2.1.36, "Offset 80h: DRAM_FERR - DRAM First Error Register").

Table 16-99. Offset A0h: DRAM_SECF_ADD - DRAM First Single Bit Error Correct Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0b	RO
30 :02	FSEFADD	**First Correctable Error Address:** This field contains system address bits 34:06 for the first correctable error (DRAM_FERR). This field is set by hardware, and represents a physical address. This field can only be reset by a PWRGD reset.	Y	0000000h	RO
01 :00	Reserved	Reserved		00b	RO

16.2.1.46 Offset A4h: DRAM_DED_ADD - DRAM Double Bit Error Address Register

Captures the address of the first DED (uncorrectable non-scrub engine) error occurring in the memory system. The value in this register is only valid if the Uncorrectable Read Memory Error bit in the DRAM_FERR register or the DRAM_NERR register has been set. The bits in this register are sticky through reset.

Table 16-100. Offset A4h: DRAM_DED_ADD - DRAM Double Bit Error Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** A4h **Offset End:** A7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0b	RO
30 :02	FUERRAD	**First Uncorrectable Error Address:** This field contains address bits 34:06 for the first uncorrectable error. This field is set by hardware, and represents a physical address. This field is only reset by a PWRGD reset.	Y	0000000h	RO
01 :00	Reserved	Reserved		00b	RO

16.2.1.47 Offset A8h: DRAM_SCRB_ADD - DRAM Scrub Error Address Register

Captures the address of the first uncorrectable error encountered by the scrub engine for a periodic memory scrub. The value in this register is only valid if the Uncorrectable Scrubber Data Error bit in the DRAM_FERR register or DRAM_NERR register has been set. The bits in this register are sticky through reset.

Table 16-101.Offset A8h: DRAM_SCRB_ADD - DRAM Scrub Error Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** A8h **Offset End:** ABh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0b	RO
30 :02	SEADD	**Scrub Error Address:** This field is updated when an uncorrectable error is encountered by the periodic memory scrubber and that scrub error causes the DRAM.FERR field to be updated, i.e. if the uncorrectable scrub error is the first error after the DRAM_FERR register is cleared. This field is set by hardware and represents a physical address. The mapping is DRAM_SCRB_ADD[30:02] = system address[34:6]. This field is only reset to zero by a PWRGR reset.	Y	0000000h	RO
01 :00	Reserved	Reserved		00b	RO

16.2.1.48 Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register

Counter for SEC errors occurring for rank 0 in the memory system. The rank counters for SEC and DED errors are implemented using a leaky bucket algorithm (see Section 11.4.7, “DDR2 MR and EMR settings”). The error count returned when this register is read is not an absolute count over time, but the sum of errors during a current specified time period plus half of the accumulated errors from past time periods. When a time period expires (determined by the Spare Control register, (see Section 16.1.1.48, “Offset 90h: SPARECTL - SPARE Control Register”), the sum of the current time period accumulated errors and a value equal to half of the past accumulated errors is retained. Half of this registered error value will be added to the errors accumulated during the next time period. This method is employed, because it is not the absolute number of errors that is most interesting, but the rate that errors occur.

When this register is written, the counter holding the number of errors with the current time period is updated. Because of this described structure, reading back the register will only return the same value written if no time period has expired between the write and the read. A write to this register does clears out any error residue that may exist from past time periods. The EP80579 tracks SEC and DED events on a per-rank basis within the DIMMs installed in the system.

The bits in this register are sticky through reset.

Note: Writing this register with a value above threshold will trigger an error.

Table 16-102.Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** B0h **Offset End:** B1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SEC_DR0A	**Count of correctable errors occurring in memory:** Rank 0SEC Count.	Y	0000h	RW

16.2.1.49 Offset B2h: DRAM_DED_R0 - DRAM Rank 0 DED Error Counter Register

Counter for DED errors occurring for rank 0. The functionality of this counter is as described in Section 16.2.1.48, "Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register" but applies to DED errors. The bits in this register are sticky through reset.

Table 16-103.Offset B2h: DRAM_DED_R0 - DRAM Rank 0 DED Error Counter Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** B2h **Offset End:** B3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DED_DR0A	**Count of uncorrectable errors occurring in memory:** Rank 0DED Count.	Y	0000h	RW

16.2.1.50 Offset B4h: DRAM_SEC_R1 - DRAM Rank 1 SEC Error Counter Register

Counter for SEC errors occurring for rank 1. The functionality of this counter is described in Section 16.2.1.48, "Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register". The bits in this register are sticky through reset.

Table 16-104.Offset B4h: DRAM_SEC_R1 - DRAM Rank 1 SEC Error Counter Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** B4h **Offset End:** B5h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SEC_R1	**Count of correctable errors occurring in memory:** Rank 1SEC Count.	Y	0000h	RW

16.2.1.51 Offset B6h: DRAM_DED_R1 - DRAM Rank 1 DED Error Counter Register

Counter for DED errors occurring for rank 1. The functionality of this counter is as described in Section 16.2.1.48, "Offset B0h: DRAM_SEC_R0 - DRAM Rank 0 SEC Error Counter Register" but applies to DED errors. The bits in this register are sticky through reset.

Table 16-105.Offset B6h: DRAM_DED_R1 - DRAM Rank 1 DED Error Counter Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** B6h **Offset End:** B7h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DED_R1	**Count of uncorrectable errors occurring in memory:** Rank 1DED Count.	Y	0000h	RW

16.2.1.52 Offset C2h: THRESH_DED - DED Error Threshold Register

This register is the threshold compare value for DED errors. An Error Threshold Detect is signaled via bit 3 of DRAM_FERR or DRAM_NERR (see Section 16.2.1.36, "Offset 80h: DRAM_FERR - DRAM First Error Register" and Section 16.2.1.37, "Offset 82h: DRAM_NERR - DRAM Next Error Register") if either of the DED event counters DRAM_DED_R0 or DRAM_DED_R1 (see Section 16.2.1.49, "Offset B2h: DRAM_DED_R0 - DRAM Rank 0 DED Error Counter Register" or Section 16.2.1.51, "Offset B6h: DRAM_DED_R1 - DRAM Rank 1 DED Error Counter Register") exceeds the value programmed into this register. The bits in this register are sticky through reset. The RANKTHREX register preserves knowledge of threshold exceeded events for software (see Section 16.2.1.56, "Offset DCh: RANKTHREX - Rank Error Threshold Exceeded Register" on page 497).

Table 16-106.Offset C2h: THRESH_DED - DED Error Threshold Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** C2h **Offset End:** C3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	THRSH_DED	**DED Error Threshold:** Threshold compare value for DED errors.	Y	0000h	RW

16.2.1.53 Offset C4h: DRAM_SECF_SYNDROME - DRAM First Single Error Correct Syndrome Register

Syndrome for correctable errors occurring in the memory system. The contents of this register are set when correctable error bit (bit 0) is set in the DRAM_FERR register (see Section 16.2.1.36, "Offset 80h: DRAM_FERR - DRAM First Error Register"). Syndrome is always logged for QW0/1 or QW2/3 pairs (block), if transferring the lower half of the cache line, and logged for QW4/5 or QW6/7 if transferring the upper half of the cache line. ECC is checked ½ cacheline at a time. The syndrome logged in this register is for the lowest ordered QW pair. For example: If both QW0/1 and QW2/3 have correctable errors, the syndrome stored is for QW0/1. A syndrome indicates error when it is a non-zero value. The bits in this register are sticky through reset.

Table 16-107.Offset C4h: DRAM_SECF_SYNDROME - DRAM First Single Error Correct Syndrome Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** C4h **Offset End:** C5h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SECF_SYND	**ECC Syndrome for DRAM_FERR correctable error**: Because only hardware writes to this register, it is read-only. **SEC mode** Bits 15:00 Mem Channel	Y	0000h	RO

16.2.1.54 Offset C6h: DRAM_SECN_SYNDROME - DRAM Next Single Error Correct Syndrome Register

Syndrome for next correctable error occurring in the memory system. The contents of this register are set when correctable error bit (bit 0) is set in the DRAM_NERR register (see Section 16.2.1.37, "Offset 82h: DRAM_NERR - DRAM Next Error Register"). The bits in this register are sticky through reset.

Table 16-108.Offset C6h: DRAM_SECN_SYNDROME - DRAM Next Single Error Correct Syndrome Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** C6h **Offset End:** C7h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SECN_SYND	**ECC Syndrome for DRAM_NERR correctable error indicated by DRAM NERR register:** Because only hardware writes to this register, it is Read-Only. Details are in Section 16.2.1.53.	Y	0000h	RO

16.2.1.55 Offset C8h: DRAM_SECN_ADD - DRAM Next Single Bit Error Correct Address Register

Captures the address of the next SEC error (either normal or scrub read) occurring in the memory system. The value in this register is only valid if the Correctable Read Memory Error bit(bit 0) in the DRAM_NERR register (see Section 16.2.1.37, "Offset 82h: DRAM_NERR - DRAM Next Error Register") has been set. The bits in this register are sticky through reset.

Table 16-109.Offset C8h: DRAM_SECN_ADD - DRAM Next Single Bit Error Correct Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:0:1	**Offset Start:** C8h **Offset End:** CBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0b	RO
30 :02	RETRADD	**Next Correctable Error Address:** This field contains system address bits 35:12 for the next correctable error. This field is set by hardware when the Correctable Read Memory Error bit in the DRAM_SERR register is set. This value represents a physical address. This field can only be reset by a PWRGD reset.	Y	0000000h	RO
01 :00	Reserved	Reserved		0b	RO

16.2.1.56 Offset DCh: RANKTHREX - Rank Error Threshold Exceeded Register

Preserves knowledge of DIMM error thresholds exceeded on a per-rank basis. Software writes bits individually to 1 to clear them. Hardware sets these bits when the count of SEC or DED errors transitions from being less than or equal to the defined threshold value (see the threshold registers such as Section 16.2.1.52, "Offset C2h: THRESH_DED - DED Error Threshold Register" on page 495) to being greater than the threshold value for a particular rank. This threshold exceeded bit is cleared by software and is only rearmed once the threshold is not exceeded. In other words, if the threshold count has been exceeded and the count is still greater than the threshold, then when software clears a given indicator bit, this same bit is not automatically set again until rearmed by the time decay of the error count and the threshold exceeded event occurs again. The bits in this register are sticky through reset.

Table 16-110.Offset DCh: RANKTHREX - Rank Error Threshold Exceeded Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** DCh **Offset End:** DDh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :06	Reserved	Reserved		000h	RO
05	RANK1_DED	**Rank 1 DED threshold exceeded:** Rank 1 DED Threshold Status. This bit is sticky through reset. Software can clear this bit by writing a 1 to the bit location. 0 = Threshold not exceeded 1 = Threshold exceeded	Y	0b	RWC
04	RANK0_DED	Rank **0 DED threshold exceeded:** Rank 0 DED Threshold Status. This bit is sticky through reset. Software can clear this bit by writing a 1 to the bit location. 0 = Threshold not exceeded 1 = Threshold exceeded	Y	0b	RWC
03 :02	Reserved	Reserved		00b	RO
01	RANK1_SEC	**Rank 1 SEC threshold exceeded:** Rank 1 SEC Threshold Status. This bit is sticky through reset. Software can clear this bit by writing a 1 to the bit location. 0 = Threshold not exceeded 1 = Threshold exceeded	Y	0b	RWC
00	RANK0_SEC	**Rank 0 SEC threshold exceeded:** Rank 0 SEC Threshold Status. This bit is sticky through reset. Software can clear this bit by writing a 1 to the bit location. 0 = Threshold not exceeded 1 = Threshold exceeded	Y	0b	RWC

16.2.1.57 Offset ECh: DERRINJCTL - DRAM Error Injection Control Register

This register controls the IMCH handling of errors on incoming data streams into the IMCH core from the DRAM interface. This register enables the injection of parity errors on incoming data streams into the core. The lower 16 bits are the corresponding flip parity bits for the cacheline of data. The upper bits in the register are for the use and control of the associated flip parity bits.

The “flip on next data transfer (bit 16)” feature is not supported in the the memory controller.

DERRINJCTL is used to inject parity errors into the data returned to MCH during data reads. The other complementary function, ECCDIAG is used to inject parity errors into DDR upon data writes.

Table 16-111.Offset ECh: DERRINJCTL - DRAM Error Injection Control Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:0:1		**Offset Start:** ECh **Offset End:** EFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :19	Reserved	Reserved	N	00h	RO
18	ENDP	**Enable / Disable data poisoning for incoming data:** This bit controls whether or not the IMCH marks data as “poisoned” when a parity error is detected on incoming data from the DRAM I/F. 0 = Errors are not propagated, only good internal parity generated. 1 = Error Poisoning Enabled. Incoming data with parity errors are marked as “poisoned” before being sent on towards its destination when in either 72-bit ECC mode via the DRC register. Error Injection is possible regardless of this bit setting. DEFAULTS TO DISABLED.	N	0b	RW
17	FLIPADT	**Enable / Disable parity bits:** Flip the designated parity bits (bits 15:00) on all data transfers into the core. If a cacheline is in progress when this register is written, wait until the start of the next cacheline to flip parity bits.	N	0b	RW
16	Reserved	Reserved	N	0b	RO
15 :00	FLIPBITS	**Two bits of parity for each 64bits of data**. 16 bits of parity for a cacheline. If the parity error injection is enabled, via setting ENDP and FLIPADT, the parity bits sent back to MCH along with the cache line read data will be exclusive-or’ed with the value in this FLIPBITS register	N	0000h	RW

16.3 EDMA Registers: Bus 0, Device 1, Function 0

The EDMA registers are in Bus 0, Device 0, Function 1. Table 16-112 provides the register address map for this device and function.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Table 16-112.Bus 0, Device 1, Function 0: Summary of EDMA PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 502	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 502	5023h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 503	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 504	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 504	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 505	80h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 505	08h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 505	00h
10h	13h	"Offset 10h: EDMALBAR - EDMA Low Base Address Register" on page 506	00000000h
2Ch	2Dh	"Offset 2Ch: SVID - Subsystem Vendor Identification Register" on page 506	0000h
2Eh	2Fh	"Offset 2Eh: SID - Subsystem Identification Register" on page 507	0000h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 507	B0h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 507	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 508	01h
40h	40h	"Offset 40h: EDMACTL - EDMA Control Register" on page 508	08h
80h	83h	"Offset 80h: EDMA_FERR - EDMA First Error Register" on page 509	00000000h
84h	87h	"Offset 84h: EDMA_NERR - EDMA Next Error Register" on page 511	00000000h
88h	88h	"Offset 88h: EDMA_EMASK - EDMA Error Mask Register" on page 513	00h
A0h	A0h	"Offset A0h: EDMA_SCICMD - EDMA SCI Command Register" on page 514	00h
A4h	A4h	"Offset A4h: EDMA_SMICMD - EDMA SMI Command Register" on page 515	00h
A8h	A8h	"Offset A8h: EDMA_SERRCMD - EDMA SERR Command Register" on page 516	00h
ACh	ACh	"Offset ACh: EDMA_MCERRCMD - EDMA MCERR Command Register" on page 517	00h
B0h	B3h	"Offset B0h: MSICR - MSI Control Register" on page 518	00020005h
B4h	B7h	"Offset B4h: MSIAR - MSI Address Register" on page 519	FEE00000h
B8h	B9h	"Offset B8h: MSIDR - MSI Data Register" on page 520	0000h

16.3.1 Register Details

16.3.1.1 Offset 00h: VID - Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies a PCI device.

Table 16-113.Offset 00h: VID - Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:**	8086h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel 8086h.		8086h	RO

16.3.1.2 Offset 02h: DID - Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device.

Table 16-114.Offset 02h: DID - Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:**	5023h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IMCH EDMA controller Function 0.		5023h	RO

16.3.1.3 Offset 04h: PCICMD - PCI Command Register

Table 16-115.Offset 04h: PCICMD - PCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		00h	
10	INTxAD	**INTx Assertion DIsable:** Controls the ability of a device to generate INTx interrupt messages. This bit only applies to legacy interrupts and not MSIs. 0 = Devices are permitted to generate INTx interrupt messages. 1 = Devices are prevented from generating INTx interrupt messages.		0b	RW
09	FB2B	**Fast Back-to-Back Enable:** Not Applicable-hardwired to 0.		0b	RO
08	SERRE	**SERR Enable:** This bit is a global enable bit for Device 1 SERR messaging. The DMA does not have an SERR# signal. The DMA communicates the SERR condition by sending an SERR message. 0 = Disable. SERR message is not generated for Device 1. 1 = Enable. Generate SERR messages for specific Device 1 error conditions that are individually enabled in the EDMA_SERRCMD register. The error status is reported in the EDMA_FERR, EDMA_NERR and PCISTS registers. ***Note:*** This bit only controls SERR messaging for Device 1. Devices 0 and 2—7 have their own SERR bits to control error reporting for error conditions occurring on their respective devices. The control bits are used in a logical OR manner to enable the SERR HI message mechanism		0b	RW
07 : 02	Reserved	Reserved		00h	
01	MAE	**Memory Access Enable:** 0 = Device 1 memory space is disabled Enable access to the EDMA Controller Low Base Address Register		0b	RW
00	Reserved	Reserved		0b	

16.3.1.4 Offset 06h: PCISTS - PCI Status Register

Table 16-116.Offset 06h: PCISTS - PCI Status Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:1:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	Reserved	Reserved		0b	
14	SSE	**Signaled System Error:** 0 = Device 1 has not generated a SERR. Software clears this bit by writing a 1 to the bit location. 1 = Indicates Device 1 generated an SERR message for any enabled Device 2 error condition. Device 1 error conditions are enabled in the PCICMD and EDMA_SERRCMD registers. Device 1 error flags are read/reset from the PCISTS, EDMA_FERR, or EDMA_NERR registers.		0b	RWC
13 : 05	Reserved	Reserved		000h	
04	CLIST	**Capability List:** This bit is set to 1 to indicate to the configuration software that this device/function implements a list of new capabilities. A list of new capabilities is accessed via register CAPPTR at configuration address offset 34h.		1b	RO
03	INTx	**NTx Status:** Indicates that an INTx interrupt is pending internal to the device. The interrupt assertion disable bit has no affect on the setting of this bit. This bit is not set for an MSI. 0 = An INTx interrupt is NOT pending 1 = An INTx interrupt is pending internal to the device		0b	RO
02 : 00	Reserved	Reserved		0h	

16.3.1.5 Offset 08h: RID - Revision Identification Register

This register contains the revision number of Device 1.

Table 16-117.Offset 08h: RID - Revision Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:1:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This is an 8-bit value that indicates the revision identification number for Device 1. This number must always be the same as the RID for Device 0, Function 0.		Variable	RO

16.3.1.6 Offset 0Ah: SUBC - Sub-Class Code Register

Table 16-118.Offset 0Ah: SUBC - Sub-Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:**	80h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SUBC	**Sub-Class Code:** This is an 8-bit value that indicates the category of other (non-specific) system peripheral.		80h	RO

16.3.1.7 Offset 0Bh: BCC - Base Class Code Register

Table 16-119.Offset 0Bh: BCC - Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:**	08h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BASEC	**Base Class Code:** This is an 8-bit value that indicates the Base Class Code for a system peripheral.		08h	RO

16.3.1.8 Offset 0Eh: HDR - Header Type Register

Table 16-120.Offset 0Eh: HDR - Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header:** This value indicates the header type of the IMCH Device 1. 00h = single-function device.		00h	RO

16.3.1.9 Offset 10h: EDMALBAR - EDMA Low Base Address Register

Table 16-121.Offset 10h: EDMALBAR - EDMA Low Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	UPBITS	**Upper Programmable Base Address:** These bits are part of the System Memory MMR region, normally set by configuration software to locate the base address of the region.		00000h	RW
11 : 04	LOWBITS	**Lower Bits:** These bits are hardwired to 0. This forces the size of the memory region to be 4 Kbyte.		000h	RO
03	PF	**Prefetchable:** This bit is hardwired to 0 to indicate that the System Memory MMR region is NON-Prefetchable.		0b	RO
02 : 01	TYPE	**Addressing Type:** These bits determine the addressing type and they are hardwired to 00 to indicate that the address range defined by the upper bits of this register can be located anywhere in the 32-bit address space as per the PCI specification for base address registers.		00b	RO
00	MSPACE	**Memory Space Indicator:** This bit is hardwired to 0 to identify the MMR range as a memory range as per the specification for PCI base address registers.		0b	RO

16.3.1.10 Offset 2Ch: SVID - Subsystem Vendor Identification Register

This value is used to identify the vendor of the subsystem.

Table 16-122.Offset 2Ch: SVID - Subsystem Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBVID	**Subsystem Vendor ID:** This field must be programmed during boot-up to indicate the vendor of the system board. When any byte or combination of bytes of this register is written, the register value locks and cannot be further updated.		0000h	RWO

16.3.1.11 Offset 2Eh: SID - Subsystem Identification Register

This value is used to identify a particular subsystem.

Table 16-123.Offset 2Eh: SID - Subsystem Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBID	**Subsystem ID:** This field must be programmed during BIOS initialization. When any byte or combination of bytes of this register is written, the register value locks and cannot be further updated.		0000h	RWO

16.3.1.12 Offset 34h: CAPPTR - Capabilities Pointer Register

The CAPPTR provides the offset that is the pointer to the location where the first set of capabilities registers is located.

Table 16-124.Offset 34h: CAPPTR - Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** B0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CAP_PTR	**Capabilities Pointer:** Pointer to first capabilities structure.		B0h	RO

16.3.1.13 Offset 3Ch: INTRLINE - Interrupt Line Register

Table 16-125.Offset 3Ch: INTRLINE - Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTRLINE	**Interrupt Connection:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller is connected with this device.		00h	RW

16.3.1.14 Offset 3Dh: INTRPIN - Interrupt Pin Register

Table 16-126.Offset 3Dh: INTRPIN - Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:**	01h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTRPIN	**Interrupt Pin:** Set to 01h to specify that EDMA always uses INTA# as its interrupt pin.		01h	RO

16.3.1.15 Offset 40h: EDMACTL - EDMA Control Register

This register defines global operation of the EDMA channels.

Table 16-127.Offset 40h: EDMACTL - EDMA Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 40h **Offset End:** 40h	
Size: 8 bit	**Default:**	08h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	EDMAEN	**EDMA Engine Enable:** 0 = EDMA Engine is disabled. 1 = EDMA Engine is enabled to do transfers. Whether the bit is set or clear, the registers can be programmed.		0b	RW
06 : 04	Reserved	Reserved		000b	RO
03	Reserved	This bit MUST be set to 1 by BIOS for proper operation.		1b	
02 : 01	Reserved	Reserved		00b	
00	PARITYEN	**Disable abort on data parity error:** 0 = Controller aborts on data parity error 1 = Controller does not abort on data parity errors, but logs the event in the appropriate EDMA FERR/NERR bit		0b	RW

16.3.1.16 Offset 80h: EDMA_FERR - EDMA First Error Register

This register captures the first occurrence of errors on a EDMA channel basis. This register only designates which channel had an error. Once an error has been captured for a given channel, this register is locked and needs to be written with ones to clear it. All DMA errors are considered non-fatal because they cause the DMA engine to stop further processing, thus avoiding any data corruption. The errors are fatal to the process, but not to the system.

Table 16-128.Offset 80h: EDMA_FERR - EDMA First Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 80h **Offset End:** 83h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Channel_3_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 3. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
30	Channel_3_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 3. (NON-FATAL)	Y	0b	RWC
29	Channel_3_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 3. (NON-FATAL)	Y	0b	RWC
28	Reserved	Reserved		0b	
27	Channel_3_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 3. (NON-FATAL)	Y	0b	RWC
26	Reserved	Reserved		0b	
25	Channel_3_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 3. (NON-FATAL)	Y	0b	RWC
24	Channel_3_Write_Error	Received write to RO descriptor registers for DMA channel 3. (NON-FATAL)	Y	0b	RWC
23	Channel_2_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 2. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
22	Channel_2_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 2. (NON-FATAL)	Y	0b	RWC
21	Channel_2_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 2. (NON-FATAL)	Y	0b	RWC
20	Reserved	Reserved		0b	
19	Channel_2_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 2. (NON-FATAL)	Y	0b	RWC
18	Reserved	Reserved		0b	
17	Channel_2_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 2. (NON-FATAL)	Y	0b	RWC
16	Channel_2_Write_Error	Received write to RO descriptor registers for DMA channel 2. (NON-FATAL)	Y	0b	RWC
15	Channel_1_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 1. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
14	Channel_1_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 1. (NON-FATAL)	Y	0b	RWC
13	Channel_1_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 1. (NON-FATAL)	Y	0b	RWC
12	Reserved	Reserved		0b	

Table 16-128.Offset 80h: EDMA_FERR - EDMA First Error Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0		**Offset Start:** 80h **Offset End:** 83h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11	Channel_1_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 1. (NON-FATAL)		Y	0b	RWC
10	Reserved	Reserved			0b	
09	Channel_1_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 1. (NON-FATAL)		Y	0b	RWC
08	Channel_1_Write_Error	Received write to RO descriptor registers for DMA channel 1. (NON-FATAL)		Y	0b	RWC
07	Channel_0_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 0. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)		Y	0b	RWC
06	Channel_0_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 0. (NON-FATAL)		Y	0b	RWC
05	Channel_0_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 0. (NON-FATAL)		Y	0b	RWC
04	Reserved	Reserved			0b	
03	Channel_0_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 0. (NON-FATAL)		Y	0b	RWC
02	Reserved	Reserved			0b	
01	Channel_0_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 0. (NON-FATAL)		Y	0b	RWC
00	Channel_0_Write_Error	Received write to RO descriptor registers for DMA channel 0. (NON-FATAL)		Y	0b	RWC

16.3.1.17 Offset 84h: EDMA_NERR - EDMA Next Error Register

This register captures EDMA channel errors after the FERR register is locked. This register accumulates all subsequent errors for the EDMA channels. See Table 16-129 for bit definitions.

Table 16-129.Offset 84h: EDMA_NERR - EDMA Next Error Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Channel_3_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 3. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
30	Channel_3_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 3. (NON-FATAL)	Y	0b	RWC
29	Channel_3_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 3. (NON-FATAL)	Y	0b	RWC
28	Reserved	Reserved		0b	
27	Channel_3_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 3. (NON-FATAL)	Y	0b	RWC
26	Reserved	Reserved		0b	
25	Channel_3_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 3. (NON-FATAL)	Y	0b	RWC
24	Channel_3_Write_Error	Received write to RO descriptor registers for DMA channel 3. (NON-FATAL)	Y	0b	RWC
23	Channel_2_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 2. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
22	Channel_2_NDAR_Alignmnt_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 2. (NON-FATAL)	Y	0b	RWC
21	Channel_2_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 2. (NON-FATAL)	Y	0b	RWC
20	Reserved	Reserved		0b	
19	Channel_2_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 2. (NON-FATAL)	Y	0b	RWC
18	Reserved	Reserved		0b	
17	Channel_2_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 2. (NON-FATAL)	Y	0b	RWC
16	Channel_2_Write_Error	Received write to RO descriptor registers for DMA channel 2. (NON-FATAL)	Y	0b	RWC
15	Channel_1_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 1. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
14	Channel_1_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 1. (NON-FATAL)	Y	0b	RWC
13	Channel_1_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 1. (NON-FATAL)	Y	0b	RWC
12	Reserved	Reserved		0b	

Table 16-129.Offset 84h: EDMA_NERR - EDMA Next Error Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11	Channel_1_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 1. (NON-FATAL)	Y	0b	RWC
10	Reserved	Reserved		0b	
09	Channel_1_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 1. (NON-FATAL)	Y	0b	RWC
08	Channel_1_Write_Error	Received write to RO descriptor registers for DMA channel 1. (NON-FATAL)	Y	0b	RWC
07	Channel_0_NDAR_Addressing_Error	The descriptor pointer in the next descriptor address register is of incorrect type or range for DMA channel 0. This includes above TOM, not in a memory range, and above available address space. (NON-FATAL)	Y	0b	RWC
06	Channel_0_NDAR_Alignment_Error	The descriptor pointer in the next descriptor address register is not aligned to an eight double-word boundary for DMA channel 0. (NON-FATAL)	Y	0b	RWC
05	Channel_0_Source_Address_Error	The source address does not comply with the source type or range for DMA channel 0. (NON-FATAL)	Y	0b	RWC
04	Reserved	Reserved		0b	
03	Channel_0_Destination_Address_Error	The destination address does not comply with the destination type or range for DMA channel 0. (NON-FATAL)	Y	0b	RWC
02	Reserved	Reserved		0b	
01	Channel_0_Parity_Error	Data parity Error in reading source data from system memory for DMA channel 0. (NON-FATAL)	Y	0b	RWC
00	Channel_0_Write_Error	Received write to RO descriptor registers for DMA channel 0. (NON-FATAL)	Y	0b	RWC

16.3.1.18 Offset 88h: EDMA_EMASK - EDMA Error Mask Register

This register masks the unit errors from being recognized and therefore not logged at the unit or global level and no interrupt/messages are generated. All channels are expected to use the same reporting structure, so only one 8-bit register is implemented.

Table 16-130.Offset 88h: EDMA_EMASK - EDMA Error Mask Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** 88h **Offset End:** 88h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	DSCPEM	**Descriptor Address Type/Range Error Mask:** Mask bit for error bit 7, 15, 23, and 31 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow descriptor address type/range error logging and signaling. 1 = Mask descriptor address type/range error logging and signaling.	Y	0b	RW
06	DSCPAE	**Descriptor Address Alignment Error:** Mask bit for error bit 6, 14, 22, and 30 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow descriptor address alignment error logging and signaling. 1 = Mask descriptor address alignment error logging and signaling.	Y	0b	RW
05	SRCEM	**Source Address Type/Range Error:** Mask bit for error bit 5, 13, 21, and 29 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow source address type/range error logging and signaling. 1 = Mask source address type/range error logging and signaling.	Y	0b	RW
04	Reserved	Reserved	Y	0b	
03	DESTEM	**Destination Address Type/Range Error:** Mask bit for error bit 3, 11, 19, and 27 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow destination address type/range error logging and signaling. 1 = Mask destination address type/range error logging and signaling.	Y	0b	RW
02	Reserved	Reserved	Y	0b	
01	MDPARERR	**Memory Data Parity Error:** Mask bit for error bit 1, 9, 17, and 25 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow memory data parity error logging and signaling. 1 = Mask memory data parity error logging and signaling.	Y	0b	RW
00	IWERR	**Illegal Write Error:** Mask bit for error bit 0, 8, 16, and 24 of EDMA_FERR and EDMA_NERR. This bit is sticky through reset. 0 = Allow illegal write error logging and signaling. 1 = Mask illegal write error logging and signaling.	Y	0b	RW

16.3.1.19 Offset A0h: EDMA_SCICMD - EDMA SCI Command Register

This register enables various errors to generate an SCI special cycle to the IICH. When an error flag is set in the EDMA_FERR or EDMA_NERR registers, it generates an SERR, SMI, or SCI special cycle when enabled in the SERRCMD, SMICMD, or SCICMD registers, or a MCERR# on the FSB when enabled in the MCERRCMD, respectively. Note that only one message type can be enabled. All channels are expected to use the same reporting structure, so only one 8-bit register is implemented.

Table 16-131.Offset A0h: EDMA_SCICMD - EDMA SCI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** A0h **Offset End:** A0h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	SCI_DSCPERR	**Descriptor Address Type/Range Error SCI Enable:** Generate SCI if bit 7, 15, 23, or 31 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
06	SCI_DSCPAE	**Descriptor Address Alignment Error SCI Enable:** Generate SCI if bit 6, 14, 22, or 30 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
05	SCI_SRCERR	**Source Address Type/Range Error SCI Enable:** Generate SCI if bit 5, 13, 21, or 29 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
04	Reserved	Reserved		0b	
03	SCI_DESTERR	**Destination Address Type/Range Error SCI Enable:** Generate SCI if bit 3, 11, 19, or 27 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
02	Reserved	Reserved		0b	
01	SCI_MDPE1	**Memory Data Parity Error SCI Enable:** Generate SCI if bit 1, 9, 17, or 25 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
00	SCI_IWE	**Illegal Write Error SCI Enable:** Generate SCI if bit 0, 8, 16, or 24 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW

16.3.1.20 Offset A4h: EDMA_SMICMD - EDMA SMI Command Register

This register enables various errors to generate an SMI special cycle to the IICH. When an error flag is set in the EDMA_FERR or EDMA_NERR registers, it generates an SERR, SMI, or SCI special cycle when enabled in the SERRCMD, SMICMD, or SCICMD registers, or a MCERR# on the FSB when enabled in the MCERRCMD, respectively. Note that only one message type can be enabled. All channels are expected to use the same reporting structure, so only one 8-bit register is implemented.

Table 16-132.Offset A4h: EDMA_SMICMD - EDMA SMI Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** A4h **Offset End:** A4h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	SMI_DSCPERR	**Descriptor Address Type/Range Error SMI Enable:** Generate SMI if bit 7, 15, 23, or 31 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
06	SMI_DSCPAE	**Descriptor Address Alignment Error SMI Enable:** Generate SMI if bit 6, 14, 22, or 30 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
05	SMI_SRCERR	**Source Address Type/Range Error SMI Enable:** Generate SMI if bit 5, 13, 21, or 29 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
04	Reserved	Reserved		0b	
03	SMI_DSTERR	**Destination Address Type/Range Error SMI Enable:** Generate SMI if bit 3, 11, 19, or 27 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
02	Reserved	Reserved		0b	
01	SMI_MDPE1	**Memory Data Parity Error SMI Enable:** Generate SMI if bit 1, 9, 17, or 25 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
00	SMI_IWE	**Illegal Write Error SMI Enable:** Generate SMI if bit 0, 8, 16, or 24 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW

16.3.1.21 Offset A8h: EDMA_SERRCMD - EDMA SERR Command Register

This register enables various errors to generate an SERR special cycle to the IICH. When an error flag is set in the EDMA_FERR or EDMA_NERR registers, it generates an SERR, SMI, or SCI special cycle when enabled in the SERRCMD, SMICMD, or SCICMD registers, or a MCERR# on the FSB when enabled in the MCERRCMD, respectively. Note that only one message type can be enabled. All channels are expected to use the same reporting structure, so only one 8-bit register is implemented.

Table 16-133.Offset A8h: EDMA_SERRCMD - EDMA SERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** A8h **Offset End:** A8h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	SERR_DSCPERR	**Descriptor Address Type/Range Error SERR Enable:** Generate SERR if bit 7, 15, 23, or 31 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
06	SERR_DSCPAE	**Descriptor Address Alignment Error SERR Enable:** Generate SERR if bit 6, 14, 22, or 30 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
05	SERR_SRCERR	**Source Address Type/Range Error SERR Enable:** Generate SERR if bit 5, 13, 21, or 29 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
04	Reserved	Reserved		0b	
03	SERR_DSTERR	**Destination Address Type/Range Error SERR Enable:** Generate SERR if bit 3, 11, 19, or 27 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
02	Reserved	Reserved		0b	
01	SERR_MDPE1	**Memory Data Parity Error SERR Enable:** Generate SERR if bit 1, 9, 17, or 25 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
00	SERR_IWE	**Illegal Write Error SERR Enable:** Generate SERR if bit 0, 8, 16, or 24 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW

16.3.1.22 Offset ACh: EDMA_MCERRCMD - EDMA MCERR Command Register

This register enables various errors to generate the MCERR# signal on the FSB. When an error flag is set in the EDMA_FERR or EDMA_NERR registers, it generates an SERR, SMI, or SCI special cycle when enabled in the SERRCMD, SMICMD, or SCICMD registers, or a MCERR# on the FSB when enabled in the MCERRCMD, respectively. Note that only one message type can be enabled. All channels are expected to use the same reporting structure, so only one 8-bit register is implemented.

Table 16-134.Offset ACh: EDMA_MCERRCMD - EDMA MCERR Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** ACh **Offset End:** ACh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MCERR_N_DSCPERR	**Descriptor Address Type/Range Error MCERR# Enable:** Generate MCERR# if bit 7, 15, 23, or 31 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
06	MCERR_N_DSCPAE	**Descriptor Address Alignment Error MCERR# Enable:** Generate MCERR# if bit 6, 14, 22, or 30 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
05	MCERR_N_SRCERR	**Source Address Type/Range Error MCERR# Enable:** Generate MCERR# if bit 5, 13, 21 or 29 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
04	Reserved	Reserved		0b	
03	MCERR_N_DSTERR	**Destination Address Type/Range Error MCERR# Enable:** Generate MCERR# if bit 3, 11, 19, or 27 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
02	Reserved	Reserved		0b	
01	MCERR_N_MDPE1	**Memory Data Parity Error MCERR# Enable:** Generate MCERR# if bit 1, 9, 17, or 25 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW
00	MCERR_N_IWE	**Illegal Write Error MCERR# Enable:** Generate MCERR# if bit 0, 8, 16, or 24 is set in any of the EDMA_FERR or EDMA_NERR registers. 0 = Disable 1 = Enable		0b	RW

16.3.1.23 Offset B0h: MSICR - MSI Control Register

The EDMA controller generates an upstream interrupt message using Message Signaled Interrupts (MSI) to the processor, bypassing the IOxAPIC. The MSI is generated by a memory write to address 0FEEx_xxxxh. The MSI Control Register (MSICR), MSI Address Register (MSIAR) and MSI Data Register (MSIDR) support this mechanism. The default values of these registers are compatible with the default value of IOxAPIC. System software can reprogram these values, if required.

The MSI Control Register (MSICR) contains all the information related to the capability of EDMA MSI interrupts.

Table 16-135.Offset B0h: MSICR - MSI Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** B0h **Offset End:** B3h	
Size: 32 bit	**Default:**	00020005h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved.		00h	
23	ADDCPBL	**Indicates 64-bit Address Capable:** Hardwired to 0 to indicate that the EDMA Controller is capable of 32-bit MSI addressing only.		0b	RO
22 : 20	MME	**Multiple Message Enable:** The software writes to this field to indicate the number of allocated messages, which is aligned to a power of two. The value programmed into this field must be less than or equal to the number requested in the Multiple Messages Capable field. When MSI is enabled, the software allocates at least one message to the device. If two MSI messages are enabled, Message 0 is used for normal interrupts, and Message 1 is used for abort/error interrupts. If only one MSI message is enabled, Message 0 is used for both normal and error interrupts.		0h	RW
19 : 17	MMC	**Multiple Message Capable:** Hardwired to a value of 001b to indicate that the EDMA requests a capability for two messages.		001b	RO
16	MSIE	**MSI Enable:** Interrupts are generated for the conditions as described in the descriptor control register for each channel. If none of these conditions are selected, software must poll for status since no interrupts of either type are generated. 0 = Legacy interrupts are generated. 1 = MSI is generated.		0b	RW
15 : 08	NXT_PTR	**Next Pointer:** Pointer to the next item in the capabilities list. Hardwired to 00h to indicate that MSI is the last item in the Capabilities List.		00h	RO
07 : 00	CAP_ID	**Capability ID:** Hardwired to 05h to indicate that the EDMA Controller is MSI capable.		05h	RO

16.3.1.24 Offset B4h: MSIAR - MSI Address Register

The MSI Address Register (MSIAR) contains all the address related information to route MSI interrupts.

Table 16-136.Offset B4h: MSIAR - MSI Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** B4h **Offset End:** B7h	
Size: 32 bit	**Default:**	FEE00000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	MSIADD	**Address:** Most significant 12 bits of the 32-bit address.		FEEh	RW
19 : 12	MSIDID	**Destination ID:** Should reflect the 63:56 bits of IOxAPIC redirection table entry.		00h	RW
11 : 04	MSIEDID	**Extended Destination ID:** Should reflect the 55:48 bits of IOxAPIC redirection table entry.		00h	RW
03	RDRCTID	**Redirection Hint:** Allows the interrupt message to be redirected. 0 = Direct. Message is delivered to the agent listed in bits 19:12 1 = Redirect. Message is delivered to an agent with a lower interrupt priority. This can be derived from bits 10:08 of the Data field		0b	RW
02	DSTMD	**Destination Mode:** Used only if the Redirection Hint bit is set to 1. 0 = Physical 1 = Logical		0b	RW
01 : 00	Reserved	Reserved.		00b	

16.3.1.25 Offset B8h: MSIDR - MSI Data Register

The MSI Data Register (MSIDR) contains all the data-related information to route MSI interrupts.

Table 16-137.Offset B8h: MSIDR - MSI Data Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:1:0	**Offset Start:** B8h **Offset End:** B9h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	TRGMD	**Trigger Mode:** Software must set this to be the same as the corresponding bit in the I/O Redirection Table for that interrupt. 0 = Edge 1 = Level		0b	RW
14	DLVSTS	**Delivery Status:** If using edge-triggered interrupts, this is always 1, since only the assertion is sent. If using level-triggered interrupts, then this bit indicates the state of the interrupt input.		0b	RW
13 : 12	Reserved	Reserved		00b	
11	DSTNMD	**Destination Mode:** Software must set this to be the same as bit 2 of MSIAR. 0 = Physical 1 = Logical		0b	RW
10 : 08	DLVMD	**Delivery Mode:** Software must set this to be the same as the corresponding bits in the I/O Redirection Table for that interrupt. 000 Fixed 001 Lowest Priority 010 SMI/PMI 011 Reserved 100 NMI 101 INIT 110 Reserved 111 ExtINT		0h	RW
07 : 00	INTRPTV	**Interrupt Vector:** Software must set this to be the same as the corresponding bits in the I/O Redirection Table for that interrupt.		00h	RW

16.4 PCI Express* Port A Standard and Enhanced Registers: Bus 0, Devices 2 and 3, Function 0

Bus 0, Device 2, Function 0 is the PCI Express* Port A (in x8 mode) or port A0 (in x4 mode) virtual PCI-to-PCI bridge. The registers described here include both the standard configuration space and the enhanced configuration space (starting at offset 100h).

Device 3 is the PCI Express* Port A1 virtual PCI-to-PCI bridge. Device 2 is PCI Express* Port A (in x8 mode) or A0 (in x4 mode). Port A1's associated PCI Express* link has a maximum lane width of x4. When Device 2 is configured as a x8 PCI Express* link, device 3 is not available. The registers described here include both the standard configuration space and the enhanced configuration space (starting at offset 100h). Except for the registers listed below, all registers for Device 3 are exactly the same as for Device 2.

Note: All registers are located in the Core power well.

Note: For platforms not using PCIe should also disable PCI Express* logic per Section 9.2.1.1, "Low power SKU with PCI Express ports removed"

Table 16-138.Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 1 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 527	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 527	5024h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 528	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 530	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 531	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 532	04h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 532	06h
0Ch	0Ch	"Offset 0Ch: CLS - Cache Line Size Register" on page 533	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 533	01h
18h	18h	"Offset 18h: PBUSN - Primary Bus Number Register" on page 534	00h
19h	19h	"Offset 19h: SBUSN - Secondary Bus Number Register" on page 534	00h
1Ah	1Ah	"Offset 1Ah: SUBUSN: Subordinate Bus Number Register" on page 535	00h
1Ch	1Ch	"Offset 1Ch: IOBASE - I/O Base Address Register" on page 535	F0h
1Dh	1Dh	"Offset 1Dh: IOLIMIT - I/O Limit Address Register" on page 536	00h
1Eh	1Fh	"Offset 1Eh: SECSTS - Secondary Status Register" on page 536	0000h
20h	21h	"Offset 20h: MBASE - Memory Base Address Register" on page 538	FFF0h
22h	23h	"Offset 22h: MLIMIT - Memory Limit Address Register" on page 539	0000h
24h	25h	"Offset 24h: PMBASE - Prefetchable Memory Base Address Register" on page 540	FFF1h
26h	27h	"Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register" on page 540	0001h
28h	28h	"Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register" on page 541	0Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register" on page 541	00h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 542	50h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 542	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 543	01h
3Eh	3Eh	"Offset 3Eh: BCTRL - Bridge Control Register" on page 543	00h
44h	44h	"Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register" on page 545	00h
45h	45h	"Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register" on page 546	00h
46h	46h	"Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register" on page 547	00h
47h	47h	"Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register" on page 547	00h
48h	48h	"Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register" on page 548	00h
50h	50h	"Offset 50h: PMCAPID - Power Management Capabilities Structure Register" on page 548	01h
51h	51h	"Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register" on page 549	58h
52h	53h	"Offset 52h: PMCAPA - Power Management Capabilities Register" on page 549	C822h
54h	55h	"Offset 54h: PMCSR - Power Management Status and Control Register" on page 550	0000h
56h	56h	"Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register" on page 551	00h

Table 16-138.Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 2 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
58h	58h	"Offset 58h: MSICAPID - MSI Capabilities Structure Register" on page 551	05h
59h	59h	"Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register" on page 552	64h
5Ah	5Bh	"Offset 5Ah: MSICAPA - MSI Capabilities Register" on page 553	0002h
5Ch	5Fh	"Offset 5Ch: MSIAR - MSI Address for PCI Express Register" on page 553	FEE00000h
60h	61h	"Offset 60h: MSIDR - MSI Data Register" on page 554	0000h
64h	64h	"Offset 64h: PEACAPID - PCI Express Features Capabilities ID Register" on page 555	10h
65h	65h	"Offset 65h: PEANPTR - PCI Express Next Capabilities Pointer Register" on page 556	00h
66h	67h	"Offset 66h: PEACAPA - PCI Express Features Capabilities Register" on page 556	0041h
68h	6Bh	"Offset 68h: PEADEVCAP - PCI Express Device Capabilities Register" on page 557	00000001h
6Ch	6Dh	"Offset 6Ch: PEADEVCTL - PCI Express Device Control Register" on page 558	0000h
6Eh	6Fh	"Offset 6Eh: PEADEVSTS - PCI Express Device Status Register" on page 560	0000h
70h	73h	"Offset 70h: PEALNKCAP - PCI Express Link Capabilities Register" on page 561	0203E481h
74h	75h	"Offset 74h: PEALNKCTL - PCI Express Link Control Register" on page 562	0001h
76h	77h	"Offset 76h: PEALNKSTS - PCI Express Link Status Register" on page 564	1001h
78h	7Bh	"Offset 78h: PEASLTCAP - PCI Express Slot Capabilities Register" on page 565	00000000h
7Ch	7Dh	"Offset 7Ch: PEASLTCTL - PCI Express Slot Control Register" on page 568	01C0h
7Eh	7Fh	"Offset 7Eh: PEASLTSTS - PCI Express Slot Status Register" on page 569	0040h
80h	83h	"Offset 80h: PEARPCTL - PCI Express Root Port Control Register" on page 570	00000000h
84h	87h	"Offset 84h: PEARPSTS - PCI Express Root Port Status Register" on page 571	00000000h
100h	103h	"Offset 100h: ENHCAPST - Enhanced Capability Structure Register" on page 571	00010001h
104h	107h	"Offset 104h: UNCERRSTS - Uncorrectable Error Status Register" on page 572	00000000h
108h	10Bh	"Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register" on page 574	00000000h
10Ch	10Fh	"Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register" on page 575	00062010h
110h	113h	"Offset 110h: CORERRSTS - Correctable Error Status Register" on page 576	00000000h
114h	117h	"Offset 114h: CORERRMSK - Correctable Error Mask Register" on page 578	00000000h
118h	11Bh	"Offset 118h: AERCACR - Advanced Error Capabilities and Control Register" on page 579	00000000h
11Ch	11Fh	"Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register" on page 580	00000000h
120h	123h	"Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register" on page 580	00000000h
124h	127h	"Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register" on page 581	00000000h
128h	12Bh	"Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register" on page 581	00000000h
12Ch	12Fh	"Offset 12Ch: RPERRCMD - Root (Port) Error Command Register" on page 582	00000000h
130h	133h	"Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register" on page 583	00000000h
134h	137h	"Offset 134h: ERRSID - Error Source ID Register" on page 585	00000000h
140h	143h	"Offset 140h: PEAUNITERR - PCI Express Unit Error Register" on page 586	00000000h
144h	147h	"Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register" on page 588	0000E000h
148h	14Bh	"Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register" on page 589	00000000h

Table 16-138.Bus 0, Device 2, Function 0: Summary of PCI Express Port A Standard and Enhanced PCI Configuration Registers (Sheet 3 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
14Ch	14Fh	"Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register" on page 591	00000000h
150h	153h	"Offset 150h: COREDMASK - Correctable Error Detect Mask Register" on page 592	00000000h
158h	15Bh	"Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register" on page 594	00000000h
160h	163h	"Offset 160h: PEAFERR - PCI Express First Error Register" on page 595	00000000h
164h	167h	"Offset 164h: PEANERR - PCI Express Next Error Register" on page 597	00000000h
168h	16Bh	"Offset 168h: PEAERRINJCTL - Error Injection Control Register" on page 597	00000000h

Table 16-139.Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 1 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor Identification Register" on page 527	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 528	5025h
04h	05h	"Offset 04h: PCICMD - PCI Command Register" on page 528	0000h
06h	07h	"Offset 06h: PCISTS - PCI Status Register" on page 530	0010h
08h	08h	"Offset 08h: RID - Revision Identification Register" on page 531	Variable
0Ah	0Ah	"Offset 0Ah: SUBC - Sub-Class Code Register" on page 532	04h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 532	06h
0Ch	0Ch	"Offset 0Ch: CLS - Cache Line Size Register" on page 533	00h
0Eh	0Eh	"Offset 0Eh: HDR - Header Type Register" on page 533	01h
18h	18h	"Offset 18h: PBUSN - Primary Bus Number Register" on page 534	00h
19h	19h	"Offset 19h: SBUSN - Secondary Bus Number Register" on page 534	00h
1Ah	1Ah	"Offset 1Ah: SUBUSN: Subordinate Bus Number Register" on page 535	00h
1Ch	1Ch	"Offset 1Ch: IOBASE - I/O Base Address Register" on page 535	F0h
1Dh	1Dh	"Offset 1Dh: IOLIMIT - I/O Limit Address Register" on page 536	00h
1Eh	1Fh	"Offset 1Eh: SECSTS - Secondary Status Register" on page 536	0000h
20h	21h	"Offset 20h: MBASE - Memory Base Address Register" on page 538	FFF0h
22h	23h	"Offset 22h: MLIMIT - Memory Limit Address Register" on page 539	0000h
24h	25h	"Offset 24h: PMBASE - Prefetchable Memory Base Address Register" on page 540	FFF1h
26h	27h	"Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register" on page 540	0001h
28h	28h	"Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register" on page 541	0Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register" on page 541	00h
34h	34h	"Offset 34h: CAPPTR - Capabilities Pointer Register" on page 542	50h
3Ch	3Ch	"Offset 3Ch: INTRLINE - Interrupt Line Register" on page 542	00h
3Dh	3Dh	"Offset 3Dh: INTRPIN - Interrupt Pin Register" on page 543	01h
3Eh	3Eh	"Offset 3Eh: BCTRL - Bridge Control Register" on page 543	00h

Table 16-139.Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 2 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
44h	44h	"Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register" on page 545	00h
45h	45h	"Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register" on page 546	00h
46h	46h	"Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register" on page 547	00h
47h	47h	"Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register" on page 547	00h
48h	48h	"Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register" on page 548	00h
50h	50h	"Offset 50h: PMCAPID - Power Management Capabilities Structure Register" on page 548	01h
51h	51h	"Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register" on page 549	58h
52h	53h	"Offset 52h: PMCAPA - Power Management Capabilities Register" on page 549	C822h
54h	55h	"Offset 54h: PMCSR - Power Management Status and Control Register" on page 550	0000h
56h	56h	"Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register" on page 551	00h
58h	58h	"Offset 58h: MSICAPID - MSI Capabilities Structure Register" on page 551	05h
59h	59h	"Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register" on page 552	64h
5Ah	5Bh	"Offset 5Ah: MSICAPA - MSI Capabilities Register" on page 553	0002h
5Ch	5Fh	"Offset 5Ch: MSIAR - MSI Address for PCI Express Register" on page 553	FEE00000h
60h	61h	"Offset 60h: MSIDR - MSI Data Register" on page 554	0000h
64h	64h	"Offset 64h: PEACAPID - PCI Express Features Capabilities ID Register" on page 555	10h
65h	65h	"Offset 65h: PEANPTR - PCI Express Next Capabilities Pointer Register" on page 556	00h
66h	67h	"Offset 66h: PEACAPA - PCI Express Features Capabilities Register" on page 556	0041h
68h	6Bh	"Offset 68h: PEADEVCAP - PCI Express Device Capabilities Register" on page 557	00000001h
6Ch	6Dh	"Offset 6Ch: PEADEVCTL - PCI Express Device Control Register" on page 558	0000h
6Eh	6Fh	"Offset 6Eh: PEADEVSTS - PCI Express Device Status Register" on page 560	0000h
70h	73h	"Offset 70h: PEA1LNKCAP - PCI Express Link Capabilities Register" on page 561	0303E441h
74h	75h	"Offset 74h: PEALNKCTL - PCI Express Link Control Register" on page 562	0001h
76h	77h	"Offset 76h: PEALNKSTS - PCI Express Link Status Register" on page 564	1001h
78h	7Bh	"Offset 78h: PEA1SLTCAP - PCI Express Slot Capabilities Register" on page 566	00000000h
7Ch	7Dh	"Offset 7Ch: PEASLTCTL - PCI Express Slot Control Register" on page 568	01C0h
7Eh	7Fh	"Offset 7Eh: PEASLTSTS - PCI Express Slot Status Register" on page 569	0040h
80h	83h	"Offset 80h: PEARPCTL - PCI Express Root Port Control Register" on page 570	00000000h
84h	87h	"Offset 84h: PEARPSTS - PCI Express Root Port Status Register" on page 571	00000000h
100h	103h	"Offset 100h: ENHCAPST - Enhanced Capability Structure Register" on page 571	00010001h
104h	107h	"Offset 104h: UNCERRSTS - Uncorrectable Error Status Register" on page 572	00000000h
108h	10Bh	"Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register" on page 574	00000000h
10Ch	10Fh	"Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register" on page 575	00062010h
110h	113h	"Offset 110h: CORERRSTS - Correctable Error Status Register" on page 576	00000000h
114h	117h	"Offset 114h: CORERRMSK - Correctable Error Mask Register" on page 578	00000000h

Table 16-139.Bus 0, Device 3, Function 0: Summary of PCI Express Port A1 Standard and Enhanced PCI Configuration Registers (Sheet 3 of 3)

Offset Start	Offset End	Register ID - Description	Default Value
118h	11Bh	"Offset 118h: AERCACR - Advanced Error Capabilities and Control Register" on page 579	00000000h
11Ch	11Fh	"Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register" on page 580	00000000h
120h	123h	"Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register" on page 580	00000000h
124h	127h	"Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register" on page 581	00000000h
128h	12Bh	"Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register" on page 581	00000000h
12Ch	12Fh	"Offset 12Ch: RPERRCMD - Root (Port) Error Command Register" on page 582	00000000h
130h	133h	"Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register" on page 583	00000000h
134h	137h	"Offset 134h: ERRSID - Error Source ID Register" on page 585	00000000h
140h	143h	"Offset 140h: PEAUNITERR - PCI Express Unit Error Register" on page 586	00000000h
144h	147h	"Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register" on page 588	0000E000h
148h	14Bh	"Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register" on page 589	00000000h
14Ch	14Fh	"Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register" on page 591	00000000h
150h	153h	"Offset 150h: COREDMASK - Correctable Error Detect Mask Register" on page 592	00000000h
158h	15Bh	"Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register" on page 594	00000000h
160h	163h	"Offset 160h: PEAFERR - PCI Express First Error Register" on page 595	00000000h
164h	167h	"Offset 164h: PEANERR - PCI Express Next Error Register" on page 597	00000000h
168h	16Bh	"Offset 168h: PEAERRINJCTL - Error Injection Control Register" on page 597	00000000h

16.4.1 Register Details

16.4.1.1 Offset 00h: VID - Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device.

Table 16-140.Offset 00h: VID - Vendor Identification Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 00h **Offset End:** 01h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:**	8086h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification Device:** This is a 16-bit value assigned to Intel.		8086h	RO

16.4.1.2 Offset 02h: DID - Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Note that the Device ID changes for each of the PCI Express* ports, starting with 5024h for Device 2.

Table 16-141.Offset 02h: DID - Device Identification Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:**	5024h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IMCH Device 2, Function 0.		5024h	RO

16.4.1.3 Offset 02h: DID - Device Identification Register

Table 16-142.Offset 02h: DID - Device Identification Register

Description:					
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5025h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IMCH Device 3, Function 0.		5025h	RO

16.4.1.4 Offset 04h: PCICMD - PCI Command Register

Many of these bits are not applicable since the primary side of this device is not an actual PCI bus.

Table 16-143.Offset 04h: PCICMD - PCI Command Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 04h **Offset End:** 05h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved.		00h	
10	INTXD	**INTx Assertion DIsable:** Controls the ability of thePCI Express* device to assert INTx interrupts. When set, devices are prevented from asserting INTx. This bit only applies to legacy interrupts and not MSIs. Also this bit has no affect on PCI Express* messages that are converted to legacy interrupts. These are only internal, device generated interrupts. 0 = Enable INTx assertion 1 = Disable INTx assertion		0b	RW
09	FB2B	**Fast Back-to-Back Enable:** Not Applicable-hardwired to 0.		0b	RO
08	SERRE	**SERR Enable:** This bit is a global enable bit for Device SERR messaging. The IMCH does not have an SERR# signal. The IMCH communicates the SERR# condition by sending an SERR message to the IICH via NSI. 0 = No SERR message is generated by the IMCH for Device (unless enabled through enhanced configuration registers). 1 = Enable SERR, SCI, or SMI messages or asserting MCERR# for specific Device error conditions.		0b	RW
07	ADSTEP	**Address/Data Stepping:** Not applicable.		0b	RO

Table 16-143.Offset 04h: PCICMD - PCI Command Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 04h **Offset End:** 05h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
06	PERRE	**Parity Error Enable:** This bit determines the device behavior on detection of a parity error. See the *PCI Express* Interface Specification, Rev 1.0a*, for details. 0 = Parity Errors are logged in the status register, but no other action is taken. 1 = Normal action is taken upon detection of Parity Error, as well as logging.			0b	RW
05	VPS	**VGA palette snoop:** Not applicable.			0b	RO
04	MWIE	**Memory Write and Invalidate Enable:** Not applicable.			0b	RO
03	SCE	**Special Cycle Enable:** Not applicable.			0b	RO
02	BME	**Bus Master Enable:** This bit controls the PCI Express* port's ability to issue memory and I/O read/write requests on behalf of subordinate devices. MSI interrupt messages are in-band memory writes, and clearing this bit disables MSI interrupt messages. 0 = Disable. The port does not respond to any I/O or memory transaction originating on the secondary interface. 1 = Enable.			0b	RW
01	MAE	**Memory Access Enable:** Controls access to the Memory and Prefetchable memory address ranges. 0 = Disable all of device memory space 1 = Enable			0b	RW
00	IOAE	**IO Access Enable:** Controls access to the I/O address range defined in the IOBASE and IOLIMIT registers. 0 = Disable device I/O space 1 = Enable			0b	RW

16.4.1.5 Offset 06h: PCISTS - PCI Status Register

PCISTS is a 16-bit status register that reports the occurrence of error conditions associated with the primary side of the “virtual” PCI-PCI bridge embedded within the IMCH.

Table 16-144.Offset 06h: PCISTS - PCI Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 06h **Offset End:** 07h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0010h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** Parity is supported on the primary side of this device. Since the parity is not checked on the downstream side from the core, this bit can never be set. 0 = No Parity Error detected		0b	RO
14	SSE	**Signaled System Error:** Indicates whether or not a NSI SERR message was generated by this device. For the root port the fatal and non-fatal messages can be either received or virtual messages that are forwarded for reporting. 0 = SERR message not generated by this device. 1 = This device was the source of fatal or non-fatal error that has been enabled for generation of a System Error. Software clears this bit by writing a ‘1’ to the bit location.		0b	RWC
13	RMAS	**Received Master Abort Status:** Indicates whether or not this PCI Express* device received a completion with Unsupported Request Completion status. 0 = No Master Abort received. Software clears this bit by writing a ‘1’ to the bit location. 1 = Set when this PCI Express* device receives a completion with Unsupported Request Completion Status.		0b	RWC
12	RTAS	0 = **Received Target Abort Status:** Indicates whether or not this PCI Express* device received a completion with Completer Abort Completion Status. No Target Abort received. Software clears this bit by writing a ‘1’ to the bit location. 1 = Set when this PCI Express* device receives a completion with Completer Abort Completion Status.		0b	RWC
11	STAS	**Signaled Target Abort Status:** Not applicable to the primary side. 0 = This PCI Express* device has not completed a request using Completer Abort Completion Status. 1 = This PCI Express* device completed a request using Completer Abort Completion Status.		0b	RO
10 : 09	DEVT	**DEVSEL# Timing:** Not Applicable. Hardwired to 0.		00b	RO
08	DPD	0 = **Master Data Parity Error Detected:** Parity is supported on the primary side of this device. No Master Parity Error detected. Software clears this bit by writing a ‘1’ to the bit location. 1 = Set when this PCI Express* device receives a completion marked poisoned, or when this device poisons a write Request. This bit can only be set if the Parity Error Enable bit is set.		0b	RWC
07	FB2B	**Fast Back-to-Back:** Not Applicable		0b	RO

Table 16-144.Offset 06h: PCISTS - PCI Status Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 06h **Offset End:** 07h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
06	Reserved	Reserved			0b	
05	C66M	**Capable 66MHz:** Not Applicable			0b	RO
04	CAPL	**Capabilities List:** Hardwired to 1 to indicate the presence of an Extended Capability List item.			1b	RO
03	INTXS	**INTx Status:** This bit does not get set for interrupts forwarded up from downstream devices, or for messages converted to interrupts by the root port. The INTx Assertion Disable bit has no effect on the setting of this bit. This bit is not set for an MSI. 0 = An INTx interrupt is not pending internal to this device. 1 = An INTx interrupt is pending internal to this device.			0b	RO
02 : 00	Reserved	Reserved			00h	

16.4.1.6 Offset 08h: RID - Revision Identification Register

This register contains the revision number of the device.

Table 16-145.Offset 08h: RID - Revision Identification Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 08h **Offset End:** 08h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:**	Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the device 2. It is always the same as the value in Device 0 RID.			Variable	RO

16.4.1.7 Offset 0Ah: SUBC - Sub-Class Code Register

This register contains the Sub-Class Code for the device.

Table 16-146.Offset 0Ah: SUBC - Sub-Class Code Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 0Ah **Offset End:** 0Ah	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:**	04h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SUBC	**Sub-Class Code:** This value indicates the category of Bridge into which device falls.04h = PCI to PCI Bridge.			04h	RO

16.4.1.8 Offset 0Bh: BCC - Base Class Code Register

This register contains the Base Class Code of the IMCH Device 2.

Table 16-147.Offset 0Bh: BCC - Base Class Code Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 0Bh **Offset End:** 0Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:**	06h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BASEC	**Base Class Code:** This value indicates the Base Class Code for the device. 06h = Bridge device.			06h	RO

16.4.1.9 Offset 0Ch: CLS - Cache Line Size Register

This register is normally set by system firmware and OS to the system cache line size.

Table 16-148.Offset 0Ch: CLS - Cache Line Size Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 0Ch **Offset End:** 0Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CLS	**Cache Line Size:** This register is set by BIOS or OS to the system cache line size. Implemented as read-write field only for compatibility reasons. It has no effect on the device's functionality.		00h	RW

16.4.1.10 Offset 0Eh: HDR - Header Type Register

This register identifies the header layout of the configuration space.

Table 16-149.Offset 0Eh: HDR - Header Type Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:**	01h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**Header Type Register:** This value indicates the Header Type of the device. 01h = single-function device with Bridge layout.		01h	RO

16.4.1.11 Offset 18h: PBUSN - Primary Bus Number Register

This register identifies that “virtual” PCI-to-PCI bridge is connected to bus 0.

Table 16-150.Offset 18h: PBUSN - Primary Bus Number Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 18h **Offset End:** 18h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 18h **Offset End:** 18h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BUSN	**Primary Bus Number:** Configuration software typically programs this field with the number of the bus on the primary side of the bridge. Since Device 2 is an internal device and its primary bus is always 0, these bits are hardwired to 0.		00h	RO

16.4.1.12 Offset 19h: SBUSN - Secondary Bus Number Register

This register identifies the bus number assigned to the second bus side of the “virtual” PCI-to-PCI bridge (the PCI Express* connection). This number is programmed by the PCI configuration software to allow mapping of configuration cycles to a second bridge device connected to PCI Express*.

Table 16-151.Offset 19h: SBUSN - Secondary Bus Number Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 19h **Offset End:** 19h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 19h **Offset End:** 19h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BUSN	**Secondary Bus Number:** This field is programmed by configuration software with the lowest bus number of the PCI Express* port.		00h	RW

16.4.1.13 Offset 1Ah: SUBUSN - Subordinate Bus Number Register

This register is programmed by PCI configuration software to the highest numbered subordinate bus (if any) that resides below another bridge device below the secondary PCI Express* interface.

Table 16-152.Offset 1Ah: SUBUSN: Subordinate Bus Number Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 1Ah **Offset End:** 1Ah	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 1Ah **Offset End:** 1Ah	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BUSN	**Subordinate Bus Number:** This register is programmed by configuration software with the number of the highest subordinate bus that lies behind the device bridge.			00h	RW

16.4.1.14 Offset 1Ch: IOBASE - I/O Base Address Register

The IOBASE and IOLIMIT registers control the processor-to-PCI Express* I/O access routing based on the following formula:

IO_BASE =< address =< IO_LIMIT

Only the upper four bits are programmable. For the purpose of address decode address bits A[11:00] are treated as 0. Thus the bottom of the defined I/O address range is aligned to a 4 Kbyte boundary.

Table 16-153.Offset 1Ch: IOBASE - I/O Base Address Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 1Ch **Offset End:** 1Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 1Ch **Offset End:** 1Ch	
Size: 8 bit	**Default:**	F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	IOBASE	**I/O Address Base:** Corresponds to A[15:12] of the I/O addresses passed by the device bridge to PCI Express*.			Fh	RW
03 : 00	IOBM	**I/O Addressing mode:** These bits are hardwired to 0. 0h = 16-bit I/O addressing All other bit combinations are not supported.			0h	RO

16.4.1.15 Offset 1Dh: IOLIMIT - I/O Limit Address Register

This register controls the CPU to PCI Express* I/O access routing based on the following formula:

IO_BASE =< address =< IO_LIMIT

Only the upper four bits are programmable. For the purpose of address decode address bits A[11:00] are assumed to be FFFh. Thus, the top of the defined I/O address range is at the top of a 4 Kbyte aligned address block.

Table 16-154.Offset 1Dh: IOLIMIT - I/O Limit Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 1Dh **Offset End:** 1Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 1Dh **Offset End:** 1Dh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	IOLIMT	**I/O Address Limit:** Corresponds to A[15:12] of the I/O address limit of device. Devices between this upper limit and IOBASE2 are passed to PCI Express*.		0h	RW
03 : 00	IOLM	**I/O Addressing mode:** These bits are hardwired to 0. 0h = 16-bit I/O addressing All other bit combinations are not supported.		0h	RO

16.4.1.16 Offset 1Eh: SECSTS - Secondary Status Register

SECSTS is a 16-bit status register that reports the occurrence of error conditions associated with the secondary side (e.g., PCI Express* side) of the “virtual” PCI-PCI bridge embedded within IMCH.

Table 16-155.Offset 1Eh: SECSTS - Secondary Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 1Eh **Offset End:** 1Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 1Eh **Offset End:** 1Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	2DPE	**Detected Parity Error:** This bit is set by the PCI Express* Port logic when the secondary side receives a poisoned TLP, regardless of the state of the Parity Error Enable bit. Software clears this bit by writing a ‘1’ to the bit location. See the *PCI Express* Interface Specification, Rev 1.0a* for details. 0 = No parity Error detected. 1 = Parity Error Detected (poisoned TLP received).		0b	RWC

Table 16-155.Offset 1Eh: SECSTS - Secondary Status Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 1Eh **Offset End:** 1Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 1Eh **Offset End:** 1Fh	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
14	2RSE	**Received System Error:** Indicates whether or not an ERR_FATAL or ERR_NONFATAL message was received via PCI Express*. 0 = Error message not received by this device. 1 = This device received fatal or non-fatal error message via PCI Express*. Software clears this bit by writing a '1' to the bit location. This bit is not set for virtual messages.			0b	RWC
13	2RMAS	**Received Master Abort Status:** Indicates whether or not this PCI Express* device received a completion with Unsupported Request Completion status. 0 = No Master Abort received. Software clears this bit by writing a '1' to the bit location. 1 = Set when this PCI Express* device receives a completion with Unsupported Request Completion Status.			0b	RWC
12	2RTAS	**Received Target Abort Status:** Indicates whether or not this PCI Express* device received a completion with Completer Abort Completion Status. 0 = No Target Abort received. Software clears this bit by writing a '1' to the bit location. 1 = Set when this PCI Express* device receives a completion with Completer Abort Completion Status.			0b	RWC
11	STAS	**Signaled Target Abort Status:** Indicates whether or not this PCI Express* device completed a request using Completer Abort Completion Status. 0 = No Target Abort signaled. Software clears this bit by writing a '1' to the bit location. 1 = Set when this PCI Express* device completes a request using Completer Abort Completion Status.			0b	RWC
10 : 09	DEVT	**DEVSEL# Timing:** Not Applicable			00b	RO
08	DPD	**Master Data Parity Error Detected:** Parity is supported on the secondary side of this device. 0 = No Master Parity Error detected. Software clears this bit by writing a '1' to the bit location. 1 = Set when this PCI Express* device receives a completion marked poisoned, or when this device poisons a write Request. This bit can only be set if the Parity Error Enable bit is set.			0b	RWC
07	FB2B	**Fast Back-to-Back:** Hardwired to 0. Not Applicable to PCI Express*.			0b	RO
06	Reserved	Reserved			0b	RO
05	CAP66	**Capability 66 MHz:** Not Applicableto PCI Express*. Hardwired to 0.			0b	RO
04 : 00	Reserved	Reserved			00h	RO

16.4.1.17 Offset 20h: MBASE - Memory Base Address Register

This register controls the processor to PCI Express* non-prefetchable memory access routing based on the following formula:

MEMORY_BASE =< address =< MEMORY_LIMIT

Note: The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32-bit address. The bottom four bits of this register are read-only and return zeroes when read. This register must be initialized by the configuration software. For the purpose of address decode address bits A[19:00] of the Memory Base Address are assumed to be 0. Similarly, the bridge assumes that the lower 20 bits of the Memory Limit Address (A[19:00]) are F_FFFFh. Thus, the bottom of the defined memory address range are aligned to a 1 Mbyte boundary, and the top of the defined memory range is at the top of a 1 Mbyte memory block. Memory range covered by MBASE and MLIMIT registers are used to map non-prefetchable PCI Express* address ranges (typically where control/status memory-mapped I/O data structures of the graphics controller resides) and PMBASE and PMLIMIT are used to map prefetchable address ranges (typically graphics local memory). This segregation allows application of USWC space attribute to be performed in a true plug-and-play manner to the prefetchable address range for improved PCI Express* memory access performance.

Configuration software is responsible for programming all address range registers (prefetchable, non-prefetchable) with the values that provide exclusive address ranges (for example, to prevent overlap with each other and/or with the ranges covered with the main memory). There is no provision in the CMI hardware to enforce prevention of overlap and operations of the system in the case of overlap are not guaranteed.

Table 16-156.Offset 20h: MBASE - Memory Base Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 20h **Offset End:** 21h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 20h **Offset End:** 21h	
Size: 16 bit	**Default:**	FFF0h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	MBASE	**Memory Address Base:** Corresponds to A[31:20] of the lower limit of the memory range that are passed by the Device 2 bridge to PCI Express*.		FFFh	RW
03 : 00	Reserved	Reserved		0h	

16.4.1.18 Offset 22h: MLIMIT - Memory Limit Address Register

This register controls the processor to PCI Express* non-prefetchable memory access routing based on the following formula:

MEMORY_BASE =< address =< MEMORY_LIMIT

The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32-bit address. The bottom four bits of this register are read-only and return zeroes when read. This register must be initialized by configuration software. For the purpose of address decode address bits A[19:00] are assumed to be FFFFFh. Thus, the top of the defined memory address range is at the top of a 1 Mbyte aligned memory block.

Note: Memory range covered by MBASE and MLIMIT registers, are used to map non-prefetchable PCI Express* address ranges (typically where control/status memory-mapped I/O data structures of the graphics controller reside) and PMBASE and PMLIMIT are used to map prefetchable address ranges (typically graphics local memory). This segregation allows application of USWC space attribute to be performed in a true plug-and-play manner to the prefetchable address range for improved PCI Express* memory access performance.

Configuration software is responsible for programming all address range registers (prefetchable, non-prefetchable) with the values that provide exclusive address ranges; for example, to prevent overlap with each other and/or with the ranges covered with the main memory. There is no provision in the CMI hardware to enforce prevention of overlap and operations of the system in the case of overlap are not guaranteed.

Table 16-157.Offset 22h: MLIMIT - Memory Limit Address Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 22h **Offset End:** 23h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 22h **Offset End:** 23h	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	MILIMIT	**Memory Address Limit:** Corresponds to A[31:20] of the memory address that corresponds to the upper limit of the range of memory accesses that are passed by the device bridge to PCI Express*.			000h	RW
03 : 00	Reserved	Reserved			0h	

16.4.1.19 Offset 24h: PMBASE - Prefetchable Memory Base Address Register

This PMBASE and PMLIMIT register controls the processor to PCI Express* prefetchable memory accesses. The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32-bit address. For the purpose of address decode, bits A[19:00] of the Prefetchable Memory Base Address are assumed to be 0. similarly, the bridge assumes that the lower 20 bits of the Prefetchable Memory Limit Address (A[19:00]) are F_FFFFh. Thus, the bottom of the defined memory address range are aligned to a 1 Mbyte boundary, and the top of the defined memory range are at the top of a 1 Mbyte memory block.

The bottom 4 bits of both the Prefetchable Memory Base and Prefetchable Memory Limit registers are read-only, contain the same value, and encode whether or not the bridge supports 64-bit addresses. If these four bits have the value 0h, then the bridge supports only 32 bit addresses. If these four bits have the value 01h, then the bridge supports 64-bit addresses and the Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers hold the rest of the 64-bit prefetchable base and limit addresses respectively. All other encodings are reserved.

Table 16-158.Offset 24h: PMBASE - Prefetchable Memory Base Address Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 24h **Offset End:** 25h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 24h **Offset End:** 25h	
Size: 16 bit	**Default:**	FFF1h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	PMBASE	**Prefetchable Memory Address Base:** Corresponds to A[31:20] of the lower limit of the address range passed by bridge device across PCI Express*.			FFFh	RW
03 : 01	MAMB	**Memory Addressing Mode.** These bits are read-only with a value of zero, all other values are reserved.			0h	RO
00	MBUAE	**Memory Base Upper Address Enabled:** 0 = Disabled 1 = Enabled - Indicates that the base address is further defined by the upper address bits of the memory base upper address register.			1h	RO

16.4.1.20 Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register

This register controls the processor to PCI Express* prefetchable memory accesses. The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32-bit address. For the purpose of address decode, bits A[19:00] are assumed to be FFFFFh. Thus, the top of the defined memory address range are at the top of a 1 Mbyte aligned memory block.

Table 16-159.Offset 26h: PMLIMIT - Prefetchable Memory Limit Address Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 26h **Offset End:** 27h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 26h **Offset End:** 27h	
Size: 16 bit	**Default:**	0001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	PMLIMIT	**Prefetchable Memory Address Limit:** Corresponds to A[31:20] of the upper limit of the address range passed by bridge Device 2 across PCI Express*.			000h	RW
03 : 01	MAML	**Memory Addressing Mode.** These bits are read-only with a value of zero, all other values are reserved.			0h	RO
00	MLUAE	**Memory Limit Upper Address Enabled:** 0 = Disabled 1 = Enabled - Indicates that the limit address is further expanded/defined by the upper address bits of the memory limit upper address register.			1h	RO

16.4.1.21 Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register

These register expands the prefetchable memory base address by four bits. All other bits are reserved.

Table 16-160.Offset 28h: PMBASU - Prefetchable Memory Base Upper Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 28h **Offset End:** 28h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 28h **Offset End:** 28h	
Size: 8 bit	**Default:**	0Fh		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03 : 00	BUA	**Base Upper Address bits:** These four bits expands the prefetchable address base to 36 bits. Corresponds to A[35:32] of the lower limit of the address range passed by bridge device across the PCI Express* interface.		Fh	RW

16.4.1.22 Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register

These register expands the prefetchable memory limit address by four bits. All other bits are reserved.

Table 16-161.Offset 2Ch: PMLMTU - Prefetchable Memory Limit Upper Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 2Ch **Offset End:** 2Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 2Ch **Offset End:** 2Ch	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03 : 00	LUA	**Limit Upper Address bits:** These four bits expands the prefetchable address limit to 36 bits. Corresponds to A[35:32] of the upper limit of the address range passed by bridge device across the PCI Express* interface.		0h	RW

16.4.1.23 Offset 34h: CAPPTR - Capabilities Pointer Register

The CAPPTR provides the offset that is the pointer to the location where the first set of capabilities registers is located.

Table 16-162.Offset 34h: CAPPTR - Capabilities Pointer Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 34h **Offset End:** 34h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:**	50h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CAP_PTR	**Capabilities Pointer:** Pointer to first PCI Express* Capabilities Structure register block, which is the first of the chain of capabilities.			50h	RO

16.4.1.24 Offset 3Ch: INTRLINE - Interrupt Line Register

Table 16-163.Offset 3Ch: INTRLINE - Interrupt Line Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 3Ch **Offset End:** 3Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTRC	**Interrupt Connection:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller that connects this device.			00h	RW

16.4.1.25 Offset 3Dh: INTRPIN - Interrupt Pin Register

Table 16-164.Offset 3Dh: INTRPIN - Interrupt Pin Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:**	01h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTRP	**Interrupt Pin:** Set to '01h' to indicate PCI Express* always uses INTA# as its interrupt pin. Once this register is written, the register value locks and cannot be further updated.		01h	RWO

16.4.1.26 Offset 3Eh: BCTRL - Bridge Control Register

This register provides extensions to the PCICMD register that are specific to PCI-PCI bridges. The BCTRL provides additional control for the secondary interface (e.g. PCI Express*) and some bits that affect the overall behavior of the "virtual" PCI-PCI bridge embedded within, e.g. VGA compatible address range mapping.

Table 16-165.Offset 3Eh: BCTRL - Bridge Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 3Eh **Offset End:** 3Eh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 3Eh **Offset End:** 3Eh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0b	
06	SRESET	**Secondary Bus Reset:** 0 = No hot reset is triggered on the link for the corresponding PCI Express* port and the PCI Express* hierarchy domain subordinate to the port. 1 = Setting this bit triggers a hot reset on the link for the corresponding PCI Express* port and the PCI Express* hierarchy domain subordinate to the port. This sends the LTSSM into the Training (or Link) Control Reset state, which necessarily implies a reset to the downstream device and all subordinate devices. Once this bit has been cleared, and the minimum transmission requirement has been met, the detect state is entered by both ends of the link. Note also that a secondary bus reset does not in general reset the primary side configuration registers of the targeted PCI Express* port. This is necessary to allow software to specify special training configuration, such as entry into loopback mode.		0b	RW
05 : 04	Reserved	Reserved		0b	

Table 16-165.Offset 3Eh: BCTRL - Bridge Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 3Eh **Offset End:** 3Eh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 3Eh **Offset End:** 3Eh	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
03	VGAEN	**VGA Enable:** Controls the routing of processor initiated transactions targeting VGA compatible I/O and memory address ranges. 0 = Disable 1 = Enable ***Note:*** Only one of Device 2–3's VGAEN bits are allowed to be set. This must be enforced via software.			0b	RW
02	ISAEN	**ISA Enable:** Modifies the response by the IMCH to an I/O access issued by the processor that target ISA I/O addresses. This applies only to I/O addresses that are enabled by the IOBASE and IOLIMIT registers. 0 = All addresses defined by the IOBASE and IOLIMIT for processor I/O transactions are mapped to PCI Express*. 1 = IMCH does not forward to PCI Express* any I/O transactions addressing the last 768 bytes in each 1 Kbyte block even if the addresses are within the range defined by the IOBASE and IOLIMIT registers. Instead, these cycles are forwarded to NSI where they can be subtractively or positively claimed by the ISA bridge.			0b	RW
01	2SERRE	**SERR Enable:** This bit enables or disables forwarding of ERR_COR, ERR_NONFATAL, and ERR_FATAL messages from PCI Express* to NSI, where they can be converted into interrupts that are eventually delivered to the processor. 0 = Disable 1 = Enable			0b	RW
00	2PERRE	**Parity Error Response Enable:** Controls response to poisoned TLPs on PCI Express*. 0 = Disable 1 = Enable			0b	RW

16.4.1.27 Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register

This register is for vendor specific commands.

Hot Plug is not supported.

Note: It appears as a reserved register, except for Device 2 which implement this Hot Plug specific register.

Table 16-166.Offset 44h: VSCMD0 - Vendor Specific Command Byte 0 Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 44h **Offset End:** 44h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 44h **Offset End:** 44h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	PDPI	**Presence Detect Polarity Invert:** 0 = Presence Detect normal polarity. 1 = The polarity of the Presence Detect input received from the IOX is inverted.		0b	RW
06	APPI	**Attention Pushbutton Polarity Invert:** 0 = Attention Pushbutton normal polarity. 1 = The polarity of the Attention Pushbutton input received from the IOX is inverted.		0b	RW
05	PFPI	**Power Fault Polarity Invert:** 0 = Power Fault normal polarity. 1 = The polarity of the Power Fault input received from the IOX is inverted.		0b	RW
04	MPI	**MRL Polarity Invert:** 0 = MRL normal polarity. 1 = The polarity of the MRL input received from the IOX is inverted.		0b	RW
03	PCPI	**Power Control Polarity Invert:** 0 = Power Control normal polarity. 1 = The polarity of the Power Control input received from the IOX is inverted.		0b	RW
02	IPI	**Indicator Polarity Invert:** 0 = Attention and Power indicators normal polarity. 1 = The polarity of the Attention and Power indicators presented to the IOX is inverted.		0b	RW
01 : 00	Reserved	Reserved		0b	

16.4.1.28 Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register

Hot Plug is not supported.

This register is for vendor specific commands.

Table 16-167.Offset 45h: VSCMD1 - Vendor Specific Command Byte 1 Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 45h **Offset End:** 45h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 45h **Offset End:** 45h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0b	
03	CTOD	**Completion TO Timer Disable:** 0 = The completion Timeout Timer is enable. 1 = The completion Timeout Timer is disabled.		0b	RW
02	HGD	**Hot Plug is not supported BIOS must set this bit to '1'** **Hot plug GPE Disable:** 0 = Enables reporting of Hot Plug interrupts via the legacy GPE mechanism. 1 = Disables reporting of Hot Plug interrupts via the legacy GPE mechanism. This bit must be set when Hot Plug interrupts are to be reported via the interrupt (INTx or MSI) signaling mechanism.		0b	RW
01	TCLE	**Training Control Loopback Enable:** 0 = Disabled 1 = Enabled - If this bit is a 1 when the TS1/TS2 ordered-sets are transmitted, the "Enable Loopback" bit is set in the training control symbol		0b	RW
00	PMETOR	**PME Turn Off Request:** 0 = Cleared by hardware when the acknowledge is returned from the link. The bit is also cleared when the link layer is in the DL_down state. 1 = Set by software if link layer is in the DL_UP state.		0b	RWS

16.4.1.29 Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register

This register is for vendor specific status.

Table 16-168.Offset 46h: VSSTS0 - Vendor Specific Status Byte 0 Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 46h **Offset End:** 46h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 46h **Offset End:** 46h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved			00h	
03	SMB_Busy	This signal indicates that the bus is busy, but that this master is not involved in the traffic. 0 = Ready 1 = Busy			0b	RO
02	Reserved	Reserved			0b	
01	Reserved	1 = Reserved			0b	RO
00	Reserved	1 = Reserved			0b	RO

16.4.1.30 Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register

This register is for vendor specific status.

Table 16-169.Offset 47h: VSSTS1 - Vendor Specific Status Byte 1 Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 47h **Offset End:** 47h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 47h **Offset End:** 47h	
Size: 8 bit	**Default:**	00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved			00h	
01	LA	**Link Active:** Bit reports whether transactions are being sent or aborted by the downstream transaction control, which is determined by the "link_active" status from the link layer reflected in this status bit. 0 = link down 1 = link up			0b	RO
00	PMETOA	**PME Turn Off Acknowledge:** 0 = Software writes a 1 to this bit to clear it. The bit will also be cleared when the link layer is in the DL_down state 1 = Set by hardware when PMETOR is ON and the acknowledge is returned from the link. When this bit is set, the Turn Off request bit clears.			0b	RWC

16.4.1.31 Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register

This register is for vendor specific commands.

Table 16-170.Offset 48h: VSCMD2 - Vendor Specific Command Byte 2 Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 48h **Offset End:** 48h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 48h **Offset End:** 48h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved	Reserved		00h	
00	DPSSEN	**NOTE: This register bit must always be disabled or undefined behavior will result.** 0 = Disable 1 = Enable		0b	RW

16.4.1.32 Offset 50h: PMCAPID - Power Management Capabilities Structure Register

This register identifies the capability structure and points to the next structure.

Table 16-171.Offset 50h: PMCAPID - Power Management Capabilities Structure Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 50h **Offset End:** 50h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 50h **Offset End:** 50h	
Size: 8 bit	**Default:**	01h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CAP_ID	This field has the value 01h to identify the CAP_ID assigned by the PCI SIG for vendor dependent capability pointers.		01h	RO

16.4.1.33 Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register

This register identifies the capability structure and points to the next structure.

Table 16-172.Offset 51h: PMNPTR - Power Management Next Capabilities Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 51h **Offset End:** 51h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 51h **Offset End:** 51h	
Size: 8 bit	**Default:**	58h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	NCR	**Next Capability Pointer:** This field points to the next Capability ID in this device which is the MSI.		58h	RO

16.4.1.34 Offset 52h: PMCAPA - Power Management Capabilities Register

This register identifies the capabilities for PM.

Table 16-173.Offset 52h: PMCAPA - Power Management Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 52h **Offset End:** 53h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 52h **Offset End:** 53h	
Size: 16 bit	**Default:**	C822h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PMES_PMCAPA	**PME Support:** Identifies power states which assert PME. Bits 15, 14 and 11 must be set to '1' for PCI-PCI bridge structures representing ports on root complexes. The definition of these bits is taken from the *PCI Bus Power Management Interface Specification Revision 1.1.* bit(11) XXXX1b PME# can be asserted from D0 bit(12) XXX1Xb PME# can be asserted from D1 (IMCH does not support) bit(13) XX1XXb PME# can be asserted from D2 (IMCH does not support) bit(14) X1XXXb PME# can be asserted from D3 hot (IMCH does not support) bit(15) 1XXXXb PME# can be asserted from D3 cold (IMCH does not support) Note: D3 is not supported, default value shows incorrect support for D3		11001b	RO
10	D2S	**D2 Support:** This bit is hardwired to '0' to Indicate the power management state D2 is not supported.		0b	RO
09	D1S	**D1 Support:** This bit is hardwired to '0' to Indicate the power management state D1 is not supported.		0b	RO

Table 16-173.Offset 52h: PMCAPA - Power Management Capabilities Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 52h **Offset End:** 53h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 52h **Offset End:** 53h	
Size: 16 bit	**Default:**	C822h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08 : 06	AUXCC	**AUXC current:** AUX current required for function. Hardwired to 000b to indicate a self-powered device.		000b	RO
05	DSI	**DSI:** Device Specific Initialization is required.		1b	RO
04	Reserved	Reserved		0b	
03	PMEC	**PME Clock:** This bit is hardwired to '0' to Indicate no PCI Clock is required for PME.		0b	RO
02 : 00	VER	**Version:** Hardwired to 010b to indicate compliance with *PCI Bus Power Management Interface Specification, Rev 1.1*.		010b	RO

16.4.1.35 Offset 54h: PMCSR - Power Management Status and Control Register

This register contains the control and status bits for power management.

Table 16-174.Offset 54h: PMCSR - Power Management Status and Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 54h **Offset End:** 55h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 54h **Offset End:** 55h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PMES_PMCSR	**PME Status: This bit is** hardwired to '0' to Indicate this field is not supported by IMCH.	Y	0b	RO
14 :09	Reserved	Reserved		00b	
08	PMEE	**PME Enable:** Controls PME# assertion. This bit is sticky through reset. Writes to this field have no effect. This bit is sticky. 0 = This device does not assert PME# 1 = Enables this device to assert PME#	Y	0b	RW
07 : 02	Reserved	Reserved		00h	
01 : 00	PS	**Power State:** Since the PCI Express* bridge device supports only the D0 state, writes to this field have no effect.		00h	RO

16.4.1.36 Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register

This register identifies the capabilities for power management.

Table 16-175.Offset 56h: PMCSRBSE - Power Management Status and Control Bridge Extensions Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 56h **Offset End:** 56h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 56h **Offset End:** 56h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0b	RO
06	Reswerd	**B2/B3 Support:** Reserved		0b	RO
05 : 00	Reserved	Reserved		00h	

16.4.1.37 Offset 58h: MSICAPID - MSI Capabilities Structure Register

This register identifies the MSI capability structure.

Table 16-176.Offset 58h: MSICAPID - MSI Capabilities Structure Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 58h **Offset End:** 58h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 58h **Offset End:** 58h	
Size: 8 bit	**Default:**	05h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CAP_ID	This field has the value 05h to identify the CAP_ID assigned by the PCI SIG for a message signaled interrupts capability list.		05h	RO

16.4.1.38 Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register

This register points to the next capability structure.

Table 16-177.Offset 59h: MSINPTR - MSI Next Capabilities Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 59h **Offset End:** 59h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 59h **Offset End:** 59h	
Size: 8 bit	**Default:**	64h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MSI_NCP	**Next Capability Pointer:** This field points to the next Capability ID in this device, which is the Hot Plug Controller.		64h	RO

16.4.1.39 Offset 5Ah: MSICAPA - MSI Capabilities Register

The PCI Express* controller generates upstream interrupt message using MSI to the processor bypassing IOxAPIC. The MSI is generated by a Memory Write to address 0FEEx_xxxxh. Three 32-bit registers are required in the PCI Express* controller to support this mechanism. The default values of these registers are compatible to the default value of IOxAPIC. The software can reprogram these registers to required value. The three registers are MSI Control Register (MSICR), MSI Address Register (MSIAR) and MSI Data Register (MSIDR). Depending on system requirement each PCI Express* channel can have a MSI block (provides better flexibility) or the PCI Express* controller as a whole has one MSI block and all channels raise hardware interrupts to this block.

The MSI Control Register (MSICR) contains all the information related to the capability of PCI Express* MSI interrupts. The MSICR register has been broken down into its components, MSICAPID, MSINPTR, and MSICAPA for purposes of separate register definitions.

Table 16-178.Offset 5Ah: MSICAPA - MSI Capabilities Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 5Ah **Offset End:** 5Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 5Ah **Offset End:** 5Bh	
Size: 16 bit	**Default:**	0002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	Reserved	Reserved			00h	
07	64AC	**Indicates 64-bit Address Capable:** Hardwired to '0' to indicate that the PCI Express* bridge is capable of 32-bit MSI addressing.			0b	RO
06 : 04	MME	**Multiple Message Enable:** The software writes this field to indicate the number of allocated messages, which is aligned to a power of two. When MSI is enabled, the software allocates at least one message to the device.			0h	RW
03 : 01	MMC	**Multiple Message Capable:** The PCI Express* requests a capability for two messages by initializing this field to a value of 001b.			001b	RO
00	MSIE	**MSI Enable:** Software sets this bit to select the method of interrupt delivery. If no interrupts are enabled, software must poll for status since no interrupts of either type are generated. 0 = Legacy interrupts are generated. 1 = Message Signaled Interrupts (MSI) are generated.			0b	RW

16.4.1.40 Offset 5Ch: MSIAR - MSI Address for PCI Express* Register

The MSI Address Register (MSIAR) contains all the address related information to route MSI interrupts.

Table 16-179.Offset 5Ch: MSIAR - MSI Address for PCI Express Register (Sheet 1 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 5Ch **Offset End:** 5Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 5Ch **Offset End:** 5Fh	
Size: 32 bit	**Default:**	FEE00000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	MSIA	**Address:** Most significant 12 bits of 32-bit address.			FEEh	RW
19 : 12	DESID	**Destination ID:** Should reflect the 63:56 bits of IOxAPIC redirection table entry. The IMCH may substitute other values in this field when redirecting to the System Bus.		Y	00h	RW
11 : 04	EXDID	**Extended Destination ID:** Should reflect the 55:48 bits of IOxAPIC redirection table entry.			00h	RW

Table 16-179.Offset 5Ch: MSIAR - MSI Address for PCI Express Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 5Ch **Offset End:** 5Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 5Ch **Offset End:** 5Fh	
Size: 32 bit	**Default:**	FEE00000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	RH	**Redirection Hint:** Used by the IMCH to allow the interrupt message to be redirected. 0 = Direct 1 = Redirect		0b	RW
02	DMMSIA	**Destination Mode:** Used only if Redirection Hint is set to '1'. 0 = Physical 1 = Logical		0b	RW
01 : 00	Reserved	Reserved		0b	

16.4.1.41 Offset 60h: MSIDR - MSI Data Register

The MSI Data Register (MSIDR) contains all the data elated information to route MSI interrupts.

Table 16-180.Offset 60h: MSIDR - MSI Data Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 60h **Offset End:** 61h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 60h **Offset End:** 61h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	TM	**Trigger Mode:** Same as the corresponding bit in the I/O Redirection Table for that interrupt. 0 = Edge 1 = Level		0b	RW
14	DVS	**Delivery Status:** If using edge-triggered interrupts, this is always a 1, since only assertion is sent. If using level-triggered interrupts, then this bit indicates the state of the interrupt input.		0b	RW
13 : 12	Reserved	Reserved		0b	

Table 16-180.Offset 60h: MSIDR - MSI Data Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 60h **Offset End:** 61h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 60h **Offset End:** 61h	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11	DMMSID	**Destination Mode:** Same as bit 2 of MSIAR. 0 = Physical 1 = Logical			0b	RW
10 : 08	DELM	**Delivery Mode:** Same as the corresponding bits in the I/O Redirection Table for that interrupt. 000= Fixed 100=NMI 001= Lowest Priority 101=INIT 010= SMI/PMI 110=Reserved 011= Reserved 111=ExtINT			0h	RW
07 : 00	IV	**Interrupt Vector:** Same as the corresponding bits in the I/O Redirection Table for that interrupt.			00h	RW

16.4.1.42 Offset 64h: PEACAPID - PCI Express* Features Capabilities ID Register

This register identifies the PCI Express* features capability structure.

Table 16-181.Offset 64h: PEACAPID - PCI Express Features Capabilities ID Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 64h **Offset End:** 64h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 64h **Offset End:** 64h	
Size: 8 bit	**Default:**	10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CAP_ID	This field has the value 10h to identify the CAP_ID assigned by the PCI SIG for PCI Express* capability structure.			10h	RO

16.4.1.43 Offset 65h: PEANPTR - PCI Express* Next Capabilities Pointer Register

This register identifies the next PCI Express* capability structure.

Table 16-182.Offset 65h: PEANPTR - PCI Express Next Capabilities Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 65h **Offset End:** 65h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 65h **Offset End:** 65h	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PEA_NCP	**Next Capability Pointer:** This field contains that value 00b to indicate that there are no additional capability structures.		00h	RO

16.4.1.44 Offset 66h: PEACAPA - PCI Express* Features Capabilities Register

This register identifies PCI Express* device type and associated capabilities.

Table 16-183.Offset 66h: PEACAPA - PCI Express Features Capabilities Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 66h **Offset End:** 67h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 66h **Offset End:** 67h	
Size: 16 bit	**Default:**	0041h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 14	Reserved	Reserved		00b	
13 : 09	CIMN	**Capability Interrupt Message Number:** If the function is allocated more than one MSI interrupt number, this field contains the offset between the base Message Data and the MSI Message that is generated when any of the status bits in either the Slot Status or Root Port Status registers of this capability structure are set. Hardware updates this field so that it is correct if the number of MSI Messages assigned to the device (based on the setting of the Multiple Message Enable bits in the MSI Capabilities register).		00000b	RO
08	SIMP	**Slot Implemented:** BIOS must set this bit at boot time if the PCI Express* link associated with this port is connected to a slot (as compared to being connected to a motherboard component, or being disabled). 0 = Slot not implemented. 1 = Slot implemented.		0b	RWO
07 : 04	DPT	**Device/Port Type:** Hardwired to a value of “4” hex to indicate a root port.		4h	RO
03 : 00	CAPV	**Capability Version:** Hardwired to 1h to indicate compliance with the *PCI Express* Interface Specification, Rev 1.0a*.		1h	RO

16.4.1.45 Offset 68h: PEADEVCAP - PCI Express* Device Capabilities Register

This register identifies the device capabilities for PCI Express*.

Table 16-184.Offset 68h: PEADEVCAP - PCI Express Device Capabilities Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 68h **Offset End:** 6Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:**	00000001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	Reserved	Reserved		0h	
27 : 26	CSPLS	**Captured Slot Power Limit Scale (Upstream Ports Only):** Specifies the scale used for the Slot Power Limit Value. 00b = 1.0x (25.5 – 255) 01b = 0.1x (2.55 – 25.5) 10b = 0.01x (0.255 – 2.55) 11b = 0.001x (0.0 – 0.255)		00b	RO
25 : 18	CSPLV	**Captured Slot Power Limit Value (Upstream Ports Only):** In combination with the Slot Power Limit Scale value, this register specifies the upper limit on power supplied by slot. Power limit (in watts) calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field. This value is set by the Set_Slot_Power_Limit.		00h	RO
17 : 06	Reserved	Reserved		000b	
05	ETFS	**Extended Tag Field Supported:** Hardwired to 0b, indicating 5 bits, as required for a Root port.		0b	RO
04 : 03	PFS	**Phantom Functions Supported:** Hardwired to 00b as required for Root ports, indicating that devices may implement all function numbers.		00b	RO
02 : 00	MPSS	***Note:*** **Max Payload Size Supported:** Hardwired to 001b to indicate a maximum 256B payload size. Note that this refers to an inbound payload size, since the outbound payload size is restricted to a cacheline size to a value of 64 B.		001b	RO

16.4.1.46 Offset 6Ch: PEADEVCTL - PCI Express* Device Control Register

This register PCI Express* device specific parameters.

Table 16-185.Offset 6Ch: PEADEVCTL - PCI Express Device Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 6Ch **Offset End:** 6Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 6Ch **Offset End:** 6Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
15	Reserved	Reserved		0b	
14 : 12	MRRS	**Max Read Request Size:** This field sets maximum read request size for the device as a requester. The IMCH does not generate read requests with size exceeding the set value. Defined encodings for this field are: ***Note:*** 000b=128 B100b=2 Kbyte 001b=256 B101b=4 Kbyte 010b=512 B110b=Reserved 011b=1 Kbyte111b=Reserved		000b	RW
11	ENS	**Enable No Snoop:** Permits the device to set the No Snoop bit in the Requester Attributes of transactions that do not require hardware enforced cache coherency. Even when this bit is set, the device can only set the No Snoop attribute on a transaction when the address of the transaction is not stored on any cache in the system. 0 = Disable 1 = Enable Software override on usage of the “No Snoop” attribute. The IMCH hard-wires this bit to 0, as it never issues transactions with that attribute set.		0b	RO
10	AUXPPE	**Auxiliary (AUX) Power PM Enable:** Not Applicable.		0b	RO
09	PFE	**Phantom Functions Enable:** Not Applicable.		0b	RO
08	ETFE	**Extended Tag Field Enable:** Not Applicable		0b	RO
07 : 05	MAXPS	**Max Payload Size:** This field sets maximum TLP payload size for the device. As a receiver, the device must handle TLPs as large as the set value; as transmitter, the device must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the max_payload_size supported in the device capabilities register. ***Note:*** Encodings above 256B are not supported. RW functionality is only maintained for compliance testing of all register bits. Defined encodings for this field are: 000b = 128 B 100b = 2 KByte 001b = 256 B 101b = 4 KByte 010b = 512 B 110b = Reserved 011b = 1 KByte 111b = Reserved		000b	RW
04	ERO	**Enable Relaxed Ordering:** If this bit is set the device is permitted to set the Relaxed Ordering bit in the attributes field of transactions it issues that do not require strong write ordering. Hard-wired to “0” in the IMCH, as no such transaction attributes are ever used on outbound requests. 0 = Disable 1 = Enable		0b	RO

Table 16-185.Offset 6Ch: PEADEVCTL - PCI Express Device Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 6Ch **Offset End:** 6Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 6Ch **Offset End:** 6Dh	
Size: 16 bit	**Default:**	0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
03	URRE	**Unsupported Request Reporting Enable:** This bit enables reporting of unsupported requests when set. Default is "0" with reporting disabled. Note that the reporting of error messages (ERR_CORR, ERR_NONFATAL, ERR_FATAL) received by Root Port is controlled exclusively by Root Port Control Register. 0 = Disable reporting of Unsupported Request errors 1 = Enable reporting of Unsupported Request errors			0b	RW
02	FERE	**Fatal Error Reporting Enable:** This bit controls the reporting of fatal errors. Note that the reporting of fatal errors is internal to the root. No external ERR_FATAL message is generated. PCICMD[SERRE] when set can also enable reporting of both internal and external errors to be reported. 0 = Disable fatal error reporting 1 = Enable fatal error reporting			0b	RW
01	NFERE	**Non-fatal Error Reporting Enable:** This bit controls the reporting of nonfatal errors. Note that the reporting of nonfatal errors is internal to the root. No external ERR_NONFATAL message is generated. PCICMD[SERRE] when set can also enable reporting of both internal and external errors to be reported. 0 = Disable nonfatal error reporting 1 = Enable nonfatal error reporting			0b	RW
00	CERE	**Correctable Error Reporting Enable:** This bit controls the reporting of correctable errors. Note that the reporting of correctable errors is internal to the root. No external ERR_CORR message is generated. 0 = Disable correctable error reporting 1 = Enable correctable error reporting			0b	RW

16.4.1.47 Offset 6Eh: PEADEVSTS - PCI Express* Device Status Register

This register provides information about PCI Express* device specific parameters.

Table 16-186.Offset 6Eh: PEADEVSTS - PCI Express Device Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 6Eh **Offset End:** 6Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 6Eh **Offset End:** 6Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 06	Reserved	Reserved		000h	
05	TP	**Transactions Pending:** Indicates that the device has transactions pending. 0 = Cleared by hardware only when all pending transactions (including completions for any outstanding non-posted requests on any used virtual channel) have been completed. 1 = Set by hardware to indicate that transactions are pending (including completions for any outstanding non-posted requests for all used Traffic Classes).		0b	RO
04	Reserved	Reserved		0b	
03	URS	**Unsupported Request Detected:** Indicates that an Unsupported Request has been detected. This bit is set upon Unsupported Request detection regardless of whether or not error reporting is enabled in the Device Control register. Software clears this bit by writing a '1' to the bit location. 0 = No Unsupported Request detected 1 = Unsupported Request detected		0b	RWC
02	FED	**Fatal Error Detected:** Indicates that a fatal error has been detected. This bit is set upon fatal error detection regardless of whether or not error reporting is enabled in the Device Control register. Software clears this bit by writing a '1' to the bit location. 0 = No fatal error detected 1 = Fatal error detected		0b	RWC
01	NFED	**Non-fatal Error Detected:** Indicates that a nonfatal error has been detected. This bit is set upon nonfatal error detection regardless of whether or not error reporting is enabled in the Device Control register. Software clears this bit by writing a '1' to the bit location. 0 = No nonfatal error detected 1 = Nonfatal error detected		0b	RWC
00	CED	**Correctable Error Detected:** Indicates that a correctable error has been detected. This bit is set upon correctable error detection regardless of whether or not error reporting is enabled in the Device Control register. Software clears this bit by writing a '1' to the bit location. 0 = No correctable error detected 1 = Correctable error detected		0b	RWC

16.4.1.48 Offset 70h: PEALNKCAP - PCI Express* Link Capabilities Register

This register identifies PCI Express* link specific capabilities.

Table 16-187.Offset 70h: PEALNKCAP - PCI Express Link Capabilities Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:**	0203E481h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	PN	**Port Number:** This field indicates the PCI Express* port number for the associated PCI Express* link.		02h	RO
23 : 18	Reserved	Reserved		00h	
17 : 15	Reserved	Reserved		111b	
14 : 12	Reserved	Reserved		110b	
11 : 10	ASPM	**CMI does not support optional L1 ASPM.** **Active State PM:** 01 L0s Entry Supported		01b	RO
09 : 04	MLW	**Maximum Link Width:** This field indicates the maximum width of the PCI Express* link. Device 2 reports a value of 001000b, indicating a maximum link width of x8. However, if two separate devices are connected to port A (Device 2) and port A1 (Device 3), the maximum link width for both ports is x4.		001000b	RO
03 : 00	MLS	**Maximum Link Speed:** 0001b 2.5 Gb/s supported All other settings are reserved.		0001b	RO

16.4.1.49 Offset 70h: PEA1LNKCAP - PCI Express* Link Capabilities Register

This register defines the capabilities of the link.

Table 16-188.Offset 70h: PEA1LNKCAP - PCI Express Link Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:**	0303E441h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	PN	**Port Number:** This field indicates the PCI Express* port number for the associated PCI Express* link.		03h	RO
23 : 18	Reserved	Reserved		00h	
17 : 15	Reserved	Reserved		111b	
14 : 12	Reserved	Reserved		110b	

Table 16-188.Offset 70h: PEA1LNKCAP - PCI Express Link Capabilities Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:**	0303E441h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11 : 10	Reserved	Reserved			01b	
09 : 04	MLW	**Maximum Link Width:** This field indicates the maximum width of the PCI Express* link. Device 3 reports a value of 000100b indicating a maximum link width of x4. All other encodings are reserved.			000100b	RO
03 : 00	MLS	**Maximum Link Speed:** 0001b 2.5 Gb/s supported All other settings are reserved.			0001b	RO

16.4.1.50 Offset 74h: PEALNKCTL - PCI Express* Link Control Register

This register controls PCI Express* link specific parameters.

Table 16-189.Offset 74h: PEALNKCTL - PCI Express Link Control Register (Sheet 1 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 74h **Offset End:** 75h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 74h **Offset End:** 75h	
Size: 16 bit	**Default:**	0001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	Reserved	Reserved			00h	
07	ES	**Extended Synch:** Provides external devices monitoring the link with additional time for to achieve bit and symbol lock before the link enters L0 state and resumes communication. 0 = Normal 1 = Reserved.			0b	RW
06	CCC	**Common Clock Configuration:** 0 = This component and the component at the opposite end of the link are operating with asynchronous reference clocks. 1 = This component and the component at the opposite end of the link are operating with a distributed common reference clock.			0b	RW

Table 16-189.Offset 74h: PEALNKCTL - PCI Express Link Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 74h **Offset End:** 75h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 74h **Offset End:** 75h	
Size: 16 bit	**Default:**	0001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
05	RL	**Retrain Link:** 0 = Link retraining not initiated. This bit always returns 0 when read. 1 = Link retraining initiated ***Note:*** Link retraining does not force a "Link Down" condition, it merely invokes "recovery."			0b	WO
04	LD	**Link Disable:** Disables/Enables the associated PCI Express* link. 0 = Enable 1 = Disable			0b	RW
03	RCB	**Read Request Return parameter "R" Control:** Hardwired to '0', indicating "RCB" capability of 64B. This is also known as Read Completion Boundary.			0b	RO
02	Reserved	Reserved			0b	
01 : 00	ASLPMC	**Active State Link PM Control:** Controls the level of active state power management supported on the associated PCI Express* link. Defined encodings are: 00b Disabled 01b L0s Entry Supported 10b Reserved 11b Reserved			01b	RW

16.4.1.51 Offset 76h: PEALNKSTS - PCI Express* Link Status Register

This register provides information about PCI Express* Link specific parameters.

Table 16-190.Offset 76h: PEALNKSTS - PCI Express Link Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 76h **Offset End:** 77h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 76h **Offset End:** 77h	
Size: 16 bit	**Default:**	1001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	LABS	**Link Autonomous Bandwidth Status** – This bit is Set by hardware to indicate that hardware has autonomously changed Link speed or width, without the Port transitioning through DL_Down status, for reasons other than to attempt to correct unreliable Link operation. This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was indicated as an autonomous change		0b	RW1C
14	LBMS	**Link Bandwidth Management Status** – This bit is Set by hardware to indicate that either of the following has occurred without the Port transitioning through DL_Down status: A Link retraining has completed following a write of 1b to the Retrain Link bit Note: This bit is Set following any write of 1b to the Retrain Link bit, including when the Link is in the process of retraining for some other reason. Hardware has changed Link speed or width to attempt to correct unreliable Link operation, either through an LTSSM timeout or a higher level process This bit must be set if the Physical Layer reports a speed or width change was initiated by the Downstream component that was not indicated as an autonomous change.		0b	RW1C
13	DLLLA	**Data Link Layer Link Active –** This bit indicates the status of the Data Link Control and Management State Machine. 1b = in DL_Active state 0b = not in DL_Active state		0b	RO
12	SCC	**Slot Clock Configuration:** This bit indicates that the component uses the same physical reference clock that the platform provides on the connector.The Read function is only allowed after Software/BIOS initialized the bit.. 0 = The component in the slot uses an independent reference clock, irrespective of the presence of a reference on the connector. 1 = The component in the slot uses the same physical reference clock provided on the connector.		1b	RWO
11	LT	**Link Training:** This read-only bit indicates that Link training is in progress (Physical Layer LTSSM in Configuration or Recovery state); hardware clears this bit once Link Training is successfully trained to the L0 state. 0 = Cleared by hardware once link training is complete 1 = Set by hardware when link training is in progress		0b	RO

Table 16-190.Offset 76h: PEALNKSTS - PCI Express Link Status Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 76h **Offset End:** 77h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 76h **Offset End:** 77h	
Size: 16 bit	**Default:**	1001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
10	Reserved	**Reserved:**		0b	RO
09 : 04	NW	**Negotiated Width:** Note that reset value is reserved, and this field remains undefined until bit 11 (Link Training) has been cleared by hardware. If training never completes, this field remains undefined. 000001b = x1 000100b = x4 001000b = x8 All other encodings are reserved		00h	RO
03 : 00	LS	**Link Speed:** Value of 1h indicates 2.5 GBytes/s link.		1h	RO

16.4.1.52 Offset 78h: PEASLTCAP - PCI Express* Slot Capabilities Register

This register identifies PCI Express* slot specific capabilities. Hot Plug is not supported.

Table 16-191.Offset 78h: PEASLTCAP - PCI Express Slot Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 78h **Offset End:** 7Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 19	PSN	**Physical Slot Number:** This hardware initialized field indicates the physical slot number attached to this port. This field must be hardware initialized to a value that assigns a slot number that is globally unique within the chassis. This field must be initialized to 0 for ports connected to devices that are either integrated on the motherboard.		000h	RWO
18 : 17	Reserved	Reserved		00b	
16 : 15	SPLS	**Slot Power Limit Scale:** Specifies the scale used for the Slot Power Limit Value. 00b = 1.0X (25.5–255) 01b = 0.1X (2.55–25.5) 10b = 0.01X (0.255–2.55) 11b = 0.001X (0.0–0.255)		00b	RWO
14 : 07	SPLV	**Slot Power Limit Value:** In combination with the Slot Power Limit Scale value, this register specifies the upper limit on power supplied by slot. Power limit (in watts) calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field. This field must be programmed at boot. Writing to this field triggers a Set_Slot_Power_Limit inband PCI Express* message.		00h	RWO

Table 16-191.Offset 78h: PEASLTCAP - PCI Express Slot Capabilities Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 78h **Offset End:** 7Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	HOTPC	**Hot Plug is not supported** BIOSmust set to 0.		0b	RWO
05	HOTPS	**Hot Plug Surprise:** Not supported. .		0b	RO
04	PIP	Hot Plug not supported. BIOSmust set to 0		0b	RWO
03	AIP	Hot Plug not supported. BIOSmust set to 0		0b	RWO
02	MSP	Hot Plug not supported. BIOSmust set to 0		0b	RWO
01	PCP	Hot Plug not supported. BIOSmust set to 0		0b	RWO
00	ABP	Hot Plug not supported. BIOSmust set to 0		0b	RWO

16.4.1.53 Offset 78h: PEA1SLTCAP - PCI Express* Slot Capabilities Register

This register identifies PCI Express* slot specific capabilities.

Note: Hot Plug is not supported.

Table 16-192.Offset 78h: PEA1SLTCAP - PCI Express Slot Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 78h **Offset End:** 7Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 19	PSN	**Physical Slot Number:** This hardware initialized field indicates the physical slot number attached to this port. This field must be hardware initialized to a value that assigns a slot number that is globally unique within the chassis. This field must be initialized to 0 for ports connected to devices that are integrated on the motherboard.		000h	RWO
18 : 17	Reserved	Reserved		00b	
16 : 15	SPLS	**Slot Power Limit Scale** Specifies the scale used for the Slot Power Limit Value. 00b = 1.0X (25.5–255) 01b = 0.1X (2.55–25.5) 10b = 0.01X (0.255–2.55) 11b = 0.001X (0.0–0.255)		00b	RWO

Table 16-192.Offset 78h: PEA1SLTCAP - PCI Express Slot Capabilities Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 78h **Offset End:** 7Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14 : 07	SPLV	**Slot Power Limit Value** In combination with the Slot Power Limit Scale value, this register specifies the upper limit on power supplied by slot. Power limit (in watts) calculated by multiplying the value in this field by the value in the Slot Power Limit Scale field. This field must be programmed at boot. Writing to this field triggers a Set_Slot_Power_Limit inband PCI Express* message.		00h	RWO
06	HOTPC	**Hot Plug is not supported** **Hot Plug capable:**		0b	RO
05	HOTPS	**Hot Plug is not supported.** **Hot Plug Surprise:**		0b	RO
04	PIP	**Hot Plug is not supported** **Power Indicator Present:**		0b	RO
03	AIP	**Hot Plug is not supported** **Attention Indicator Present:**		0b	RO
02	MSP	**Hot Plug is not supported** **MRL Sensor Present:**		0b	RO
01	PCP	**Hot Plug is not supported** **Power Controller Present:** .		0b	RO
00	ABP	**Hot Plug is not supported** **Attention Button Present:**		0b	RO

16.4.1.54 Offset 7Ch: PEASLTCTL - PCI Express* Slot Control Register

This register controls PCI Express* Slot specific parameters.

Hot Plug is not supported.

Table 16-193.Offset 7Ch: PEASLTCTL - PCI Express Slot Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 7Ch **Offset End:** 7Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 7Ch **Offset End:** 7Dh	
Size: 16 bit	**Default:**	01C0h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		00h	
10	PCC	Hot Plug not supported. **Power Controller Control**		0b	RW
09 : 08	PIC	Hot Plug not supported. **Power Indicator Control**		01b	RW
07 : 06	AIC	Hot Plug not supported. **Attention Indicator Control**		11b	RW
05	HPIE	Hot Plug is not supported. **Hot Plug Interrupt Enable** BIOS must leave this bit at 0		0b	RW
04	CCIE	Hot Plug not supported. **Command Complete Interrupt Enable**		0b	RW
03	PDCIE	Hot Plug not supported. **Presence Detect Changed Interrupt Enable**.		0b	RW
02	MSCIE	Hot Plug not supported. **MRL Sensor Changed Interrupt Enable**		0b	RW
01	PFDIE	Hot Plug not supported. **Power Fault Detected Interrupt Enable**		0b	RW
00	ABDIE	Hot Plug not supported. **Attention Button Pressed Interrupt Enable:**		0b	RW

16.4.1.55 Offset 7Eh: PEASLTSTS - PCI Express* Slot Status Register

This register provides information about PCI Express* Slot specific parameters.

Hot Plug is not supported.

Table 16-194.Offset 7Eh: PEASLTSTS - PCI Express Slot Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 7Eh **Offset End:** 7Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 7Eh **Offset End:** 7Fh	
Size: 16 bit	**Default:**	0040h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 07	Reserved	Reserved		000h	
06	PDS	Hot Plug not supported. **Presence Detect State**		1b	RO
05	MSS	Hot Plug not supported. **MRL Sensor State**0 = MRL Closed 1 = MRL Open		0b	RO
04	COMC	Hot Plug not supported. **Command Completed**		0b	RWC
03	PDC	Hot Plug not supported. **Presence Detect Changed**		0b	RWC
02	MSC	Hot Plug not supported. **MRL Sensor Changed**		0b	RWC
01	PFD	Hot Plug not supported. **Power Fault Detected**		0b	RWC
00	ATBP	Hot Plug not supported. **Attention Button Pressed**		0b	RWC

16.4.1.56 Offset 80h: PEARPCTL - PCI Express* Root Port Control Register

This register enables the forwarding of error messages based on messages received.

Table 16-195.Offset 80h: PEARPCTL - PCI Express Root Port Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 80h **Offset End:** 83h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 80h **Offset End:** 83h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	Reserved	Reserved		0000000h	
03	EPI	**Enable PME Interrupt:** Enables/disables interrupt generation upon receipt of a PME message as reflected in the PME Status register bit. A PME interrupt is also generated if the PME Status register bit is already set when this bit is set from a cleared state. 0 = Disable PME interrupt (MCHPME#) generation 1 = Enable PME interrupt (MCHPME#) generation		0b	RW
02	ESEFE	**Enable System Error on Fatal Error:** Controls the Root Complex's response to fatal errors reported by any of the devices in the hierarchy associated with this Root Port. System error generation based on fatal errors also enabled by PCICMD[SERRE]. 0 = Disable System Error generation in response to fatal errors reported on this port. 1 = Enable System Error generation in response to fatal errors reported on this port.		0b	RW
01	ESENFE	**Enable System Error on Non-Fatal Error:** Controls the Root Complex's response to nonfatal errors reported by any of the devices in the hierarchy associated with this Root Port. System error generation based on non-fatal errors also enabled by PCICMD[SERRE]. 0 = Disable System Error generation in response to nonfatal errors reported on this port. 1 = Enable System Error generation in response to nonfatal errors reported on this port.		0b	RW
00	ESECE	**Enable System Error on Correctable Error:** Controls the Root Complex's response to correctable errors reported by any of the devices in the hierarchy associated with this Root Port. 0 = Disable System Error generation in response to correctable errors reported on this port. 1 = Enable System Error generation in response to correctable errors reported on this port.		0b	RW

16.4.1.57 Offset 84h: PEARPSTS - PCI Express* Root Port Status Register

This register supports power management events.

Table 16-196.Offset 84h: PEARPSTS - PCI Express Root Port Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 84h **Offset End:** 87h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 18	Reserved	Reserved		0000h	
17	PMEP	**PME Pending:** 0 = Cleared by hardware when no more PMEs are pending. 1 = Indicates that another PME is pending when the PME Status bit is set.		0b	RO
16	PMES	**PME Status:** 0 = Cleared by software writing a '1' to the bit location. 1 = PME has been asserted by the requestor indicated in the PME Requestor ID field. ***Note:*** Subsequent PMEs are kept pending until cleared by software.		0b	RWC
15 : 00	PMERID	**PME Requestor ID:** Indicates the PCI Requestor ID of the last PME requestor.		0000h	RO

16.4.1.58 Offset 100h: ENHCAPST - Enhanced Capability Structure Register

This register identifies the capability structure and points to the next structure. This enhanced configuration structure by definition starts at configuration offset 100h.

Table 16-197.Offset 100h: ENHCAPST - Enhanced Capability Structure Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 100h **Offset End:** 103h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 100h **Offset End:** 103h	
Size: 32 bit	**Default:**	00010001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	NCP	**Next Capability Pointer:** This field is hardwired to 000h to indicate that there are no other items in the capability list.		000h	RO
19 : 16	CV	**Capability Version:** Hardwired to 1h, to indicate *PCI Express* Interface Specification, Rev 1.0a*.		1h	RO
15 : 00	Extended_CAP_ID	Hardwired to 0001h, to indicate advanced error reporting capability.		0001h	RO

16.4.1.59 Offset 104h: UNCERRSTS - Uncorrectable Error Status Register

The Uncorrectable Error Status register reports the status of individual error sources on the PCI Express* device. An individual error status bit that is set indicates that a particular error occurred. Software may clear an error status bit by writing a '1' to the bit location. These bits are sticky through reset.

Table 16-198.Offset 104h: UNCERRSTS - Uncorrectable Error Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 104h **Offset End:** 107h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 104h **Offset End:** 107h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 21	Reserved	Reserved		000h	
20	USR_UNCERRSTS	**Unsupported Request** This error, if the first uncorrectable error, loads the header log. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unsupported Request detected.	Y	0b	RWC
19	EES	**ECRC Error Status.**Note: ECRC is not supported for the EP80579.	Y	0b	RO
18	MTS	**Malformed TLP Status.**This error, if the first uncorrectable error, loads the header log. Malformed TLP errors include: data payload length issues, byte enable rule violations, and various other illegal field settings. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Malformed TLP detected	Y	0b	RWC
17	ROS	**Receiver Overflow Status**Optional PCI Express* specification bit, implemented for IMCH. This error, if the first uncorrectable error, loads the header log. IMCH checks for overflows on the following upstream queues: posted, non-posted, and completion. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Overflow detected.	Y	0b	RWC
16	UCS	**Unexpected Completion Status.**This bit is set when the device receives a completion which does not correspond to any of the outstanding requests issued by that device. This error, if the first uncorrectable error, loads the header log. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Unexpected Completion detected.	Y	0b	RWC
15	CAS	**Completer Abort Status**Optional PCI Express* specification bit, implemented for IMCH. If a request received violates the specific programming model of this device, but is otherwise legal, this bit is set. This error, if the first uncorrectable error, load the header log. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completer Abort detected.	Y	0b	RWC
14	CTS	**Completion Timeout Status**The Completion Timeout timer must expire if a Request is not completed in 50 ms, but must not expire earlier than 50 µs. When the timer expires, this bit is set. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Completion timeout detected.	Y	0b	RWC

Table 16-198.Offset 104h: UNCERRSTS - Uncorrectable Error Status Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 104h **Offset End:** 107h
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 104h **Offset End:** 107h
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13	FCPES	**Flow Control Protocol Error Status**Optional PCI Express* specification bit, implemented for IMCH. 0 = Cleared by writing a '1' to the bit location. 1 = Flow Control Protocol Error detected. CMI asserts this bit for one of two conditions: • An FC update has been received which describes header or data credits for P, NP, or CPL which were originally advertised as infinite during initialization but are now advertised with non-zero or non-infinite values. • The number of credits advertised in an update is less than the number of credits in the previous update. The hardware accepts this flow control update, as it cannot determine if this update or the previous one was in error. ***Note:*** This bit is sticky through reset.	Y	0b	RWC
12	PTS	**Poisoned TLP Status**This bit when set indicates that some portion of the TLP data payload was corrupt. This error, if the first uncorrectable error, loads the header log. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Poisoned TLP detected.	Y	0b	RWC
11 : 05	Reserved	Reserved		00h	
04	DLPES	**Data Link Protocol Error Status**This bit is set when an ACK/NAK received does not specify the sequence number of an unacknowledged TLP, or of the most recently acknowledged TLP. This bit is sticky through reset. 0 = Cleared by writing a '1' to the bit location. 1 = Data Link Protocol Error detected.	Y	0b	RWC
03 : 00	Reserved	Reserved		000b	

16.4.1.60 Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register

The Uncorrectable Error Mask register controls reporting of individual errors by device to the PCI Express* Root Complex via a PCI Express* error message. A masked error (respective bit set in mask register) is not reported to the PCI Express Root Complex by an individual device. However, masked errors are still logged in the Uncorrectable Error Status register. There is one mask bit corresponding to every implemented bit in the Uncorrectable Error Status register. These bits are sticky through reset.

Table 16-199.Offset 108h: UNCERRMSK - Uncorrectable Error Mask Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 108h **Offset End:** 10Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 108h **Offset End:** 10Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 21	Reserved	Reserved		000h	
20	USR_UNCERRMSK	**Unsupported Request:** 0 = Report Unsupported Request Error 1 = Mask Unsupported Request Error	Y	0b	RW
19	EEM	**ECRC Error Mask:** Note: ECRC is not supported for the EP80579.	Y	0b	RO
18	MTM	**Malformed TLP Mask:** 0 = Report Malformed TLP Error 1 = Mask Malformed TLP Error	Y	0b	RW
17	ROM	**Receiver Overflow Mask:** Optional PCI Express* specification bit, implemented for. 0 = Report Receiver Overflow Error 1 = Mask Receiver Overflow Error	Y	0b	RW
16	UCM	**Unexpected Completion Mask :** 0 = Report Receiver Overflow Error 1 = Mask Receiver Overflow Error	Y	0b	RW
15	CAM	**Completer Abort Mask:** Optional PCI Express* specification bit, implemented for. 0 = Report Completer Abort Error 1 = Mask Completer Abort Error	Y	0b	RW
14	CTM	**Completion Timeout Mask:** 0 = Report Completion Timeout Error 1 = Mask Completion Timeout Error	Y	0b	RW
13	FCPEM	**Flow Control Protocol Error Mask:** Optional PCI Express* specification bit, implemented for. 0 = Report Flow Control Protocol Error 1 = Mask Flow Control Protocol Error	Y	0b	RW
12	PTM	**Poisoned TLP Mask:** 0 = Report Poisoned TLP Error 1 = Mask Poisoned TLP Error	Y	0b	RW
11 : 05	Reserved	Reserved		00h	
04	DLPEM	**Data Link Protocol Error Mask:** 0 = Report Data Link Protocol Error 1 = Mask Data Link Protocol Error	Y	0b	RW
03 : 00	Reserved	Reserved		000b	RO

16.4.1.61 Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register

The Uncorrectable Error Severity register controls whether an individual error is reported as a nonfatal or fatal error. An error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered nonfatal. These bits are sticky through reset.

Table 16-200.Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 10Ch **Offset End:** 10Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 10Ch **Offset End:** 10Fh	
Size: 32 bit	**Default:**	00062010h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 21	Reserved	Reserved		000h	
20	USR_UNCERRSEV	**Unsupported Request:** 0 = Nonfatal 1 = Fatal	Y	0b	RW
19	EESEV	**ECRC Error Severity:** Note: ECRC is not supported for the EP80579.	Y	0b	RO
18	MTSEV	**Malformed TLP Severity:** 0 = Nonfatal 1 = Fatal	Y	1b	RW
17	ROSEV	**Receiver Overflow Severity:** Optional PCI Express* specification bit, implemented for IMCH. 0 = Nonfatal 1 = Fatal	Y	1b	RW
16	UCSEV	**Unexpected Completion Severity:** 0 = Nonfatal 1 = Fatal	Y	0b	RW
15	CASEV	**Completer Abort Severity [STICKY]:** Optional PCI Express* specification bit, implemented for IMCH. 0 = Nonfatal 1 = Fatal	Y	0b	RW
14	CTSEV	**Completion Timeout Severity:** 0 = Nonfatal 1 = Fatal	Y	0b	RW
13	FCPESEV	**Flow Control Protocol Error Severity:** Optional PCI Express* specification bit, implemented for IMCH. 0 = Nonfatal 1 = Fatal	Y	1b	RW
12	PTSEV	**Poisoned TLP Severity:** 0 = Nonfatal 1 = Fatal	Y	0b	RW

Table 16-200.Offset 10Ch: UNCERRSEV - Uncorrectable Error Severity Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 10Ch **Offset End:** 10Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 10Ch **Offset End:** 10Fh	
Size: 32 bit	**Default:**	00062010h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 : 05	Reserved	Reserved		00h	
04	DLPESEV	**Data Link Protocol Error Severity:** 0 = Nonfatal 1 = Fatal	Y	1b	RW
03 : 00	Reserved	Reserved		000b	RO

16.4.1.62 Offset 110h: CORERRSTS - Correctable Error Status Register

The Correctable Error Status register reports the status of individual error sources on the PCI Express* device. An individual error status bit that is set indicates that a particular error occurred. Software may clear an error status bit by writing a '1' to the bit location. These bits are sticky through reset.

Table 16-201.Offset 110h: CORERRSTS - Correctable Error Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 110h **Offset End:** 113h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 110h **Offset End:** 113h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 13	Reserved	Reserved		00000h	
12	RTTS	**Replay Timer Timeout Status:** The replay timer counts time since the last ACK or NAK DLLP was received. When the timer expires, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Replay Timer timeout detected.	Y	0b	RWC
11 : 09	Reserved	Reserved		000b	
08	RNRS	**REPLAY_NUM Rollover Status:** A 2-bit counter counts the number of times the retry buffer has been retransmitted. When this counter rolls over, this bit is set. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = REPLAY_NUM rollover detected.	Y	0b	RWC

Table 16-201.Offset 110h: CORERRSTS - Correctable Error Status Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 110h **Offset End:** 113h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 110h **Offset End:** 113h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
07	BDS	**Bad DLLP Status:** This bit is set when the calculated DLLP CRC is not equal to the received value. This bit is sticky through system reset. 0 = Cleared by writing a '1' to the bit location. 1 = Bad DLLP detected.		Y	0b	RWC
06	BTS	**Bad TLP Status:** 0 = TLP status good. 1 = This bit is set when the calculated TLP CRC is not equal to the received value. Also included are invalid sequence numbers.		Y	0b	RWC
05 : 01	Reserved	Reserved			000b	
00	RES	**Receiver Error Status:** Optional PCI Express* specification bit, implemented for IMCH. Data is delivered over PCI Express* via packets built out of 8b/10b symbols. Receiver Error Status register is set for 8b/10b errors received, framing errors received irrespective of the packet boundaries. 0 = Cleared by writing a '1' to the bit location. 1 = Receiver Error detected.		Y	0b	RWC

16.4.1.63 Offset 114h: CORERRMSK - Correctable Error Mask Register

The Correctable Error Mask register controls reporting of individual errors by device to the PCI Express* Root Complex via a PCI Express* error message. A masked error (respective bit set in mask register) is not reported to the PCI Express* Root Complex by an individual device. However, masked errors are still logged in the Correctable Error Status register. There is one mask bit corresponding to every implemented bit in the Correctable Error Status register. These bits are sticky through reset.

Table 16-202.Offset 114h: CORERRMSK - Correctable Error Mask Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 114h **Offset End:** 117h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 114h **Offset End:** 117h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 13	Reserved	Reserved		00000h	
12	RTTM	**Replay Timer Timeout Mask:** This bit is sticky through system reset. 0 = Report Replay Timer Timeout error. 1 = Mask Replay Timer timeout error.	Y	0b	RW
11 : 09	Reserved	Reserved		000b	
08	RNRM	**REPLAY_NUM Rollover Mask:** This bit is sticky through system reset. 0 = Report REPLAY_NUM rollover 1 = Mask REPLAY_NUM rollover.	Y	0b	RW
07	BDM	**Bad DLLP Mask:** This bit is sticky through system reset. 0 = Report Bad DLLP error. 1 = Mask Bad DLLP error.	Y	0b	RW
06	BTM	**Bad TLP Mask:** This bit is sticky through system reset. 0 = Report Bad TLP error. 1 = Mask Bad TLP error.	Y	0b	RW
05 : 01	Reserved	Reserved		0000b	
00	REM	**Receiver Error Mask:** This bit is sticky through system reset. 0 = Report Receiver error. 1 = Mask Receiver error.	Y	0b	RW

16.4.1.64 Offset 118h: AERCACR - Advanced Error Capabilities and Control Register

This register identifies the capability structure and points to the next structure. The first error pointer rearms after the unmasked errors have been cleared. Software after clearing the errors must read the register again to ensure that it is indeed cleared. If it finds that another error occurred, it can not rely on the pointer or header, unless it detects that the error pointer changed from the last time it was read for the previous error. Bits in this register also declare the ECRC capability of this device. These bits are sticky through reset.

Table 16-203.Offset 118h: AERCACR - Advanced Error Capabilities and Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 118h **Offset End:** 11Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 118h **Offset End:** 11Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 09	Reserved	Reserved		000000h	
08	ECE	**ECRC Check Enable** ***Note:*** ECRC is not supported for the EP80579.	Y	0b	RO
07	EC	**ECRC Check Capable** ***Note:*** ECRC is not supported for the EP80579.		0b	RO
06	EGE	**ECRC Generation Enable** ***Note:*** ECRC is not supported for the EP80579.	Y	0b	RO
05	EGC	**ECRC Generation Capable** ***Note:*** ECRC is not supported for the EP80579.		0b	RO
04 : 00	FEP	**First error pointer** identifies the bit position of the first error reported in the Uncorrectable Error Status register. However, if a subsequent Uncorrectable Error occurs with a higher severity, this field is over-written with the bit position of the subsequent error status bit. Also, if multiple errors of equal severity are logged simultaneously, this field identifies the bit position of the most significant (leftmost) bit that has been set in the Uncorrectable Error Status register. In the event of simultaneous errors, the pointer indicates the least significant bit of the group. This bit is sticky through system reset.	Y	00000b	RO

16.4.1.65 Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register

This register contains the first 32 bits of the header log locked down when the first uncorrectable error occurs that saves the header. To rearm this register all report uncorrectable errors must be cleared from the register. Software after clearing the errors must read the register again to ensure that it is indeed cleared. If it finds that another error occurred, it can not rely on the pointer or header, unless it detects that the error pointer changed from the last time it was read for the previous error. Byte 0 of the header is located in byte 3 of the Header Log Register 0, byte 1 of the header is in byte 2 of the Header Log Register 0 and so forth. For 12 byte headers, only the first three of the four Header Log Registers are used, and values in HDRLOG3 are undefined. These bits are sticky through reset.

Table 16-204. Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 11Ch **Offset End:** 11Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 11Ch **Offset End:** 11Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	HL0	**Header Log 0** A masked error (respective bit set to '1' in mask register) is not logged in the Header Log Register, does not update the First Error Pointer, and is not reported to the PCI Express* Root Complex by an individual device.	Y	00000000h	RO

16.4.1.66 Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register

The function of the Header Log registers is described in Section 16.4.1.65, "Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register". Header Log DW1 contains the second 32 bits of the header. Byte 4 of the header is located in byte 3 of the Header Log Register 1, byte 5 of the header is in byte 2 of the Header Log Register 1 and so forth. These bits are sticky through reset.

Table 16-205.Offset 120h: HDRLOG1 - Header Log DW 1 (2nd 32 bits) Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 120h **Offset End:** 123h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 120h **Offset End:** 123h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	HL1	**Header Log 1** A masked error (respective bit set to '1' in mask register) is not logged in the Header Log Register, does not update the First Error Pointer, and is not reported to the PCI Express* Root Complex by an individual device. These bits are sticky through system reset.	Y	00000000h	RO

16.4.1.67 Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register

The function of the Header Log registers is described in Section 16.4.1.65, "Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register". Header Log DW2 contains the third 32 bits of the header. Byte 8 of the header is located in byte 3 of the Header Log Register 2, byte 9 of the header is in byte 2 of the Header Log Register 2 and so forth. These bits are sticky through reset.

Table 16-206.Offset 124h: HDRLOG2 - Header Log DW 2 (3rd 32 bits) Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 124h **Offset End:** 127h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 124h **Offset End:** 127h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	HL2	**Header Log 2** A masked error (respective bit set to '1' in mask register) is not logged in the Header Log Register, does not update the First Error Pointer, and is not reported to the PCI Express* Root Complex by an individual device. These bits are sticky through system reset.	Y	00000000h	RO

16.4.1.68 Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register

The function of the Header Log registers is described in Section 16.4.1.65, "Offset 11Ch: HDRLOG0 - Header Log DW 0 (1st 32 bits) Register". Header Log DW3 contains the fourth 32 bits of the header. For 16-byte headers, byte 12 of the header is located in byte 3 of the Header Log Register 3, byte 13 of the header is in byte 2 of the Header Log Register 3 and so forth. For 12 byte headers, values in this register are undefined. These bits are sticky through reset.

Table 16-207.Offset 128h: HDRLOG3 - Header Log DW 3 (4th 32 bits) Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 128h **Offset End:** 12Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 128h **Offset End:** 12Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	HL3	**Header Log 3** A masked error (respective bit set to '1' in mask register) is not logged in the Header Log Register, does not update the First Error Pointer, and is not reported to the PCI Express* Root Complex by an individual device. These bits are sticky through system reset.	Y	00000000h	RO

16.4.1.69 Offset 12Ch: RPERRCMD - Root (Port) Error Command Register

This register controls the generation of interrupts (beyond the basic root complex capability to generate system errors) upon detection of errors. System error generation in response to PCI Express* error messages may be turned off by system software using the PCI Express* Capability Structure when advanced error reporting via interrupts is enabled. These bits are sticky through reset.

Table 16-208.Offset 12Ch: RPERRCMD - Root (Port) Error Command Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 12Ch **Offset End:** 12Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 12Ch **Offset End:** 12Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 03	Reserved	Reserved		0000000h	
02	FEIE	**Fatal Error Interrupt Enable:** Enables the generation of an interrupt when a fatal error is reported by any of the devices in the hierarchy associated with this Root Port. This bit is sticky through reset. 0 = Disable interrupt generation on fatal error. 1 = Enable interrupt generation on fatal error.	Y	0b	RW
01	NEIE	**Nonfatal Error Interrupt Enable:** Enables the generation of an interrupt when an nonfatal error is reported by any of the devices in the hierarchy associated with this Root Port. This bit is sticky through reset. 0 = Disable interrupt generation on nonfatal error. 1 = Enable interrupt generation on nonfatal error.	Y	0b	RW
00	CEIE	**Correctable Error Interrupt Enable:** Enables the generation of an interrupt when a correctable error is reported by any of the devices in the hierarchy associated with this Root Port. This bit is sticky through reset. 0 = Disable interrupt generation on correctable error. 1 = Enable interrupt generation on correctable error.	Y	0b	RW

16.4.1.70 Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register

This register reports the status of errors received by the root complex. Each correctable and uncorrectable (nonfatal and fatal) error source has a First Error bit and a Next Error bit. When an error is received by the root complex, the associated First Error bit is set and the Requestor ID is logged in the Error Source Identification register. Software may clear an error status bit by writing a '1' to the bit location. If software does not clear the first reported error before another error is received, the Next Error status bit is set, but the Requestor ID of the subsequent error message is discarded. These bits are sticky through reset.

Table 16-209.Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 130h **Offset End:** 133h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 130h **Offset End:** 133h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 27	AEIMN	**Advanced Error Interrupt Message Number:** If this function has been allocated more than one MSI interrupt number, this field reflects the offset between the base Message Data and the MSI Message that is generated when any of the status bits of this capability are set.		0h	RO
26 : 07	Reserved	Reserved		000000h	
06	FEMD	**Fatal Error Messages Detected:** This bit is used by error handling software to determine whether fatal errors are outstanding in the hierarchy. In hardware, this bit along with bits 4 and 2 is used to clear fatal error escalation. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Fatal error message detected.	Y	0b	RWC
05	NFEMD	**Non-Fatal Error Messages Detected:** This bit is used by error handling software to determine whether non-fatal errors are outstanding in the hierarchy. In hardware, this bit along with bits 4 and 2 is used to clear non-fatal error escalation. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Non-fatal error message detected.	Y	0b	RWC
04	FUFF	**First Uncorrectable Fatal Flag:** This bit captures the nature of the first uncorrectable error message detected (and logged in the error source ID register). These bits are sticky through system reset. 0 = First uncorrectable error is non-fatal. 1 = First uncorrectable error is fatal. Software uses this flag to determine whether the uncorrectable error source ID belongs to the fatal or non-fatal error handler routine in the event that the two are independent.	Y	0b	RWC

Table 16-209.Offset 130h: RPERRMSTS - Root (Port) Error Message Status Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 130h **Offset End:** 133h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 130h **Offset End:** 133h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
03	MUEMD	**Multiple Uncorrectable Error Messages Detected** In the unlikely event of two first errors occurring during the same clock period, only the first uncorrectable error message bit is set. It takes an error to occur in a subsequent clock to set this bit. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Set when either a fatal or nonfatal error is received, and the First Uncorrectable Error Detected bit is already set. This indicates that one or more message Requestor IDs were lost.		Y	0b	RWC
02	FUEMD	**First Uncorrectable Error Message Detected** The Root Error Status bit reports status of error messages (ERR_NONFATAL and ERR_FATAL) received by the root complex, and of errors detected/reported (not masked) by the Root Port itself. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Set when the first fatal or nonfatal error is received.		Y	0b	RWC
01	MCEMD	**Multiple Correctable Error Messages Detected**In the unlikely event of two first errors occurring during the same clock period, only the first correctable error message bit is set. It takes an error to occur in a subsequent clock to set this bit. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Set when a correctable error is received, and the First Correctable Error Detected bit is already set. This indicates that one or more message Requestor IDs were lost.		Y	0b	RWC
00	FCEMD	**First Correctable Error Message Detected**The Root Error Status bit reports status of error messages (ERR_COR) received by the root complex, and of errors detected/reported (not masked) by the Root Port itself. These bits are sticky through system reset. 0 = Software clears this bit by writing a '1' to the bit location. 1 = Set when the first correctable error is received.		Y	0b	RWC

16.4.1.71 Offset 134h: ERRSID - Error Source ID Register

This register reports the source (Requestor ID) of the first correctable and uncorrectable (fatal or nonfatal) errors reported in the Root Error Status register. This register is updated regardless of the settings of the Root Control register and the Root Error Command register. These bits are sticky through reset.

Table 16-210.Offset 134h: ERRSID - Error Source ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 134h **Offset End:** 137h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 134h **Offset End:** 137h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	UESID	**Uncorrectable Error Source ID** Requestor ID of the source when an uncorrectable error (fatal or nonfatal) is received, and the First Uncorrectable Error Detected bit is not already set. Since this ID could be for an internally detected error or from a message received from the other end of the link, in the event of errors detected in the same clock, priority is given to the error received from the link, and that ID is what is logged. These bits are sticky through system reset.	Y	0000h	RO
15 : 00	CESID	**Correctable Error Source ID** Requestor ID of the source when an correctable error is received, and the First Correctable Error Detected bit is not already set. Since this ID could be for an internally detected error or from a message received from the other end of the link, in the event of errors detected in the same clock, priority is given to the error received from the link, and that ID is what is logged. These bits are sticky through system reset.	Y	0000h	RO

16.4.1.72 Offset 140h: PEAUNITERR - PCI Express* Unit Error Register

This register is specific to the IMCH. It captures the non-PCI Express* unit errors (those beyond the scope of the bus specification). The unit error mechanism is parallel to that used by “compatible” error registers and masks, but cannot feed back into standard registers because that would confuse standardized error handling software (which would not understand the extracurricular error bits). Escalation is controlled via the PEAERRDOCMD register (D2, F0:140-143h) for both standard and -specific error types. Uncorrectable fatal errors feed into the fatal reporting select, Uncorrectable non-fatal errors feed into the non-fatal reporting select, and correctable errors feed into the correctable reporting select. The lower nibble is for HPC related errors.

Table 16-211.Offset 140h: PEAUNITERR - PCI Express Unit Error Register (Sheet 1 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 140h **Offset End:** 143h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 140h **Offset End:** 143h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved for future additions.			0000h	
15	UPQOS	**Upstream Posted Queue Overflow Status:** This bit is one of the components of the Receiver Overflow Status bit in the UNCERRSTS register. Even though this bit can be set, it is only reported through the receiver overflow bit in the UNCERRSTS register. The setting of this bit is never logged in the local FERR/NERR registers or subsequently the global FERR/NERR registers, nor does it cause a SCI/SMI/SERR or MCERR message. At most, when the report mask, is disabled, it could affect the unit error pointer. This functionality is provided as an aid to debug. 0 = Software clears this bit by writing a '1' to the bit position. 1 = Overflow occurred for one of the posted header or data queues.		Y	0b	RWC
14	UNPQOS	**Upstream Non-Posted Queue Overflow Status:** This bit is set if an overflow occurs for the non-posted header queue. There is no upstream non-posted data queue. It is one of the components of the Receiver Overflow Status bit in the UNCERRSTS register. Even though this bit can be set, it is only reported through the receiver overflow bit in the UNCERRSTS register. The setting of this bit is never logged in the local FERR/NERR registers or subsequently the global FERR/NERR registers, nor is it a cause for a SCI/SMI/SERR or MCERR message. At most, when the report mask, is disabled, it could affect the unit error pointer. This functionality is provided as an aid to debug. 0 = Software clears this bit by writing a '1' to the bit position. 1 = Overflow occurred for the non-posted header queues.		Y	0b	RWC
13	UCQOS	**Upstream Completion Queue Overflow Status [STICKY]:** This bit is set if an overflow occurs for either the completion header or data queues It is one of the components of the Receiver Overflow Status bit in the UNCERRSTS register. Even though this bit can be set, it is only reported through the receiver overflow bit in the UNCERRSTS register. The setting of this bit is never logged in the local FERR/NERR registers or subsequently the global FERR/NERR registers, nor is it a cause a SCI/SMI/SERR or MCERR message. This functionality is provided as an aid to debug. 0 = Software clears this bit by writing a '1' to the bit position. 1 = Overflow occurred for one of the completion header or data queues.		Y	0b	RWC
12	LPE	0 = **LLE Protocol Error [STICKY]:** This bit is set when the transaction layer detects a protocol error on the receiver interface from the LLE. Such an event should cause retraining eventually, but not necessarily immediately. The transaction with a problem is dropped. Software clears this bit by writing a '1' to the bit position. 1 = Transaction layer detected a protocol error on the receiver interface from the LLE		Y	0b	RWC

Table 16-211.Offset 140h: PEAUNITERR - PCI Express Unit Error Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 140h **Offset End:** 143h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 140h **Offset End:** 143h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11	LDE	0 = **Link Down Error [STICKY]:** Software clears this bit by writing a '1' to the bit position. 1 = Set when the link transitions from DL_UP to DL_DOWN.		Y	0b	RWC
10	DDQPE	**Downstream Data Queue Parity Error [STICKY]:** A parity error occurred in the downstream data queue. 0 = Software clears this bit by writing a '1' to the bit position. 1 = Parity error occurred in the downstream data queue.		Y	0b	RWC
09	Reserved	**Reserved**			0b	
08	Reserved	Reserved			0b	
07	Reserved	Reserved			0b	
06	Reserved	Reserved			0b	
05	Reserved	Reserved.			0b	
04	CommBlkPAR	Common Block Parity Error [Sticky]: Indicates that a parity error occurred on a configuration register within the common block. 0 = No parity error 1 = Parity error occurred.		Y	0b	RWC
03	Reserved	Reserved.			0b	
02	SMBCLTO	**SMB Clock Low Timeout [STICKY]:**. 0 = Software clears this bit by writing a '1' to the bit position. 1 = SMB CLK low greater than 25 ms.		Y	0b	RWC
01	UESMBN	Unexpected NAK on SMB **[STICKY]**: 0 = Software clears this bit by writing a '1' to the bit position. 1 = Unexpected NAK on SMB detected.		Y	0b	RWC
00	SMBLA	SMB lost bus arbitration. (Correctable) **[STICKY]:** This bit is sticky through reset. 0 = Software clears this bit by writing a '1' to the bit position. 1 = SMB lost bus arbitration.		Y	0b	RWC

16.4.1.73 Offset 144h: PEAMASKERR - PCI Express* Unit Mask Error Register

This register is used for selecting the global error reporting method for the various error conditions.

Table 16-212.Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 144h **Offset End:** 147h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 144h **Offset End:** 147h	
Size: 32 bit	**Default:**	0000E000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	For future additions.		0000h	
15	UPQOM	**Upstream Posted Queue Overflow Mask:** Defaults to masked, normally reported through PCI Express* receive overflow status bit. 0 = Enable upstream posted queue overflow reporting. 1 = Disable upstream posted queue overflow reporting.	Y	1b	RW
14	UNPQOM	**Upstream Non-Posted Queue Overflow Mask:** Defaults to masked, normally reported through PCI Express* receive overflow status bit. 0 = Enable upstream non-posted queue overflow reporting. 1 = Disable upstream non-posted queue overflow reporting.	Y	1b	RW
13	UCQOM	**Upstream Completion Queue Overflow Mask:** Defaults to masked, normally reported through PCI Express* receive overflow status bit. 0 = Enable upstream completion queue overflow reporting. 1 = Disable upstream completion queue overflow reporting.	Y	1b	RW
12	LPEM	**LLE Protocol Error Mask:** 0 = Enable LLE protocol error reporting. 1 = Disable LLE protocol error reporting.	Y	0b	RW
11	LDEM	**Link Down Error Mask:** Mask reporting of detected link transitions from DL_UP to DL_DOWN. 0 = Enable link down error mask reporting. 1 = Disable link down error mask reporting.	Y	0b	RW
10	DDQPERM	**Downstream Data Queue Parity Error Reporting Mask:** 0 = Enable 1 = Disable	Y	0b	RW
09	Reserved	**Reserved**		0b	
08	Reserved	**Reserved**		0b	
07	Reserved	**Reserved**		0b	
06	Reserved	**Reserved**		0b	
05	Reserved	**Reserved**		0b	
04	CommBlk-PARRM	Common Block Parity Reporting Mask : 0 = Enable CommBlkPAR reporting. 1 = Disable CommBlkPAR reporting.	Y	0b	RW
03	Reserved	Reserved		0b	

Table 16-212.Offset 144h: PEAMASKERR - PCI Express Unit Mask Error Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 144h **Offset End:** 147h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 144h **Offset End:** 147h	
Size: 32 bit	**Default:**	0000E000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	SMBCLTORM	**SMBCLTO Reporting Mask]:** 0 = Enable SMBCLTO reporting. 1 = Disable SMBCLTO reporting.	Y	0b	RW
01	UESMBNRM	**UESMBN Reporting Mask :** 0 = Enable UESMBN reporting. 1 = Disable UESMBN reporting.	Y	0b	RW
00	SMBLARM	**SMBLA Reporting Mask :** 0 = Enable SMBLA reporting. 1 = Disable SMBLA reporting.	Y	0b	RW

16.4.1.74 Offset 148h: PEAERRDOCMD - PCI Express* Error Do Command Register

This register supports PCI Express* error commands for doing various signaling. DO_SCI, DO_SMI, and DO_MCERR, DO_SERR must further be enabled by the PCI Express* Host Do Command register.

Table 16-213.Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 148h **Offset End:** 14Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 148h **Offset End:** 14Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Reserved	Reserved		000b	
28 : 24	FEPUPCE	**First Error Pointer for unmasked PCI Express* correctable errors**This pointer is rearmed when all unmasked errors have been cleared. In the event of simultaneous errors, the pointer indicates the least significant bit of the group. These bits are sticky.	Y	00h	RO
23 : 21	Reserved	Reserved		000b	
20 : 16	FEPPE	**First Error Pointer for PCI Express*-unit errors** This pointer is locked once any units errors are logged in the PEAFERR. It is rearmed when all PEAUNIT errors have been cleared. In the event of simultaneous errors, the pointer indicates the least significant bit of the group. This pointer is only valid for an error that is enabled for reporting. These bits are sticky.	Y	00h	RO

Table 16-213.Offset 148h: PEAERRDOCMD - PCI Express Error Do Command Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 148h **Offset End:** 14Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 148h **Offset End:** 14Bh	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15	EHLULPE	**Enable HEADER LOG use for LLE PROTOCOL ERROR:** The header log is used by PCI Express* uncorrectable errors. This feature is used to capture the header log for the LLE protocol error in the unit error register during debug. 0 = Disable 1 = Enable			0b	RW
14	PURE	**PCI Express* Unit report enable:** This bit enables reporting of fatal or non-fatal or correctable unit errors. 0 = Disable 1 = Enable			0b	RW
13 : 12	PURSFE	**PCI Express* unit report steering for fatal errors:** 00b= SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW
11 : 10	PURSNFE	**PCI Express* unit report steering for non-fatal errors:** 00b=SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW
09 : 08	PURSCE	**PCI Express* unit report steering for correctable errors:** 00b=SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW
07 : 06	Reserved	Reserved			00b	
05 : 04	RPRSFE	**Root Port report steering for fatal errors:** If the System Error on Fatal Error bit in the Root Port Control register is set, all fatal root port errors are reported via SERR regardless of the setting of this register. MSI Enable takes precedence for this capability feature. 00b= SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW
03 : 02	RPRSNFE	**Root Port report steering for non-fatal errors:** If the System Error on Nonfatal Error bit in the Root Port Control register is set, all nonfatal root port errors are reported via SERR regardless of the setting of this register. MSI Enable takes precedence for this capability feature. 00b= SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW
01 : 00	RPRSCE	**Root Port report steering for correctable errors:** If the System Error on Correctable Error bit in the Root Port Control register is set, all correctable root port errors are reported via SERR regardless of the setting of this register. Note that MSI Enable takes precedence for this capability feature. 00b= SCI 10b=SERR 01b= SMI 11b=MCERR			00b	RW

16.4.1.75 Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register

The Uncorrectable Error Detect Mask register controls detection of the individual errors. An error event that is masked in this register, is treated as though the error never happened, and is subsequently not logged in the Uncorrectable Error Status register, nor is it ever reported. There is one mask bit corresponding to every implemented bit in the Uncorrectable Error Status register. This register is specific to the IMCH. These bits are sticky through reset.

Table 16-214.Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 14Ch **Offset End:** 14Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 14Ch **Offset End:** 14Fh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 21	Reserved	Reserved		00000000000b	
20	UREDM	**Unsupported Request Error Detect Mask. [STICKY]:** 0 = Detect Unsupported Request Error 1 = Disable Unsupported Request Error detection	Y	0b	RW
19	Reserved	***Note:*** Reserved	Y	0b	RO
18	MTEDM	**Malformed TLP Error Detect Mask. [STICKY]:** 0 = Detect Malformed TLP Error 1 = Disable Malformed TLP Error detection	Y	0b	RW
17	ROEDM	**Receiver Overflow Error Detect Mask. [STICKY]:** OPTIONAL 0 = Detect Receiver Overflow Error 1 = Disable Receiver Overflow Error detection	Y	0b	RW
16	UCEDM	**Unexpected Completion Error Detect Mask. [STICKY]:** 0 = Detect Unexpected Completion Error 1 = Disable Unexpected Completion Error detection	Y	0b	RW
15	CAEDM	**Completer Abort Error Detect Mask. [STICKY]:** OPTIONAL 0 = Detect Completer Abort Error 1 = Disable Completer Abort Error detection	Y	0b	RW
14	CTEDM	**Completion Timeout Error Detect Mask. [STICKY]:** 0 = Detect Completion Timeout Error 1 = Disable Completion Timeout Error detection	Y	0b	RW
13	FCPEDM	**Flow Control Protocol Error Detect Mask. [STICKY]:** OPTIONAL 0 = Detect Flow Control Protocol Error 1 = Disable Flow Control Protocol Error detection	Y	0b	RW
12	PTEDM	**Poisoned TLP Error Detect Mask. [STICKY]:** 0 = Detect Poisoned TLP Error 1 = Disable Poisoned TLP Error detection	Y	0b	RW

Table 16-214.Offset 14Ch: UNCEDMASK - Uncorrectable Error Detect Mask Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 14Ch **Offset End:** 14Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 14Ch **Offset End:** 14Fh	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
11 : 05	Reserved	Reserved			0000000b	
04	DLPEDM	**Data Link Protocol Error Detect Mask. [STICKY]:** 0 = Detect Data Link Protocol Error 1 = Disable Data Link Protocol Error detection		Y	0b	RW
03 :00	Reserved	Reserved			0000b	RO

16.4.1.76 Offset 150h: COREDMASK - Correctable Error Detect Mask Register

The Correctable Error Detect Mask register controls detection of the individual errors. An error event that is masked in this register is not logged in the Correctable Error Status register, and is never reported. There is one mask bit corresponding to every implemented bit in the Correctable Error Status register. These bits are sticky through reset.

Table 16-215.Offset 150h: COREDMASK - Correctable Error Detect Mask Register (Sheet 1 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 150h **Offset End:** 153h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 150h **Offset End:** 153h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 13	Reserved	Reserved			00000h	
12	RTTEDM	**Replay Timer Timeout Error Detect Mask.** This bit is sticky through system reset. 0 = Detect Replay Timer Timeout error. 1 = Disable Replay Timer timeout error detection.		Y	0b	RW
11 : 09	Reserved	Reserved			000b	
08	RNREDM	**REPLAY_NUM Rollover Error Detect Mask**This bit is sticky through system reset. 0 = Detect REPLAY_NUM rollover 1 = Disable REPLAY_NUM rollover detection.		Y	0b	RW
07	BDEDM	**Bad DLLP Error Detect Mask**This bit is sticky through system reset. 0 = Detect Bad DLLP error. 1 = Disable Bad DLLP error detection.		Y	0b	RW

Table 16-215.Offset 150h: COREDMASK - Correctable Error Detect Mask Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 150h **Offset End:** 153h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 150h **Offset End:** 153h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
06	BTEDM	**Bad TLP Error Detect Mask**OPTIONAL. This bit is sticky through system reset. 0 = Detect Bad TLP error. 1 = Disable Bad TLP error detection.		Y	0b	RW
05 : 01	Reserved	Reserved			0000b	
00	REDM	**Receiver Error Detect Mask**OPTIONAL. This bit is sticky through system reset. 0 = Detect Receiver error. 1 = Disable Receiver Error error detection.		Y	0b	RW

16.4.1.77 Offset 158h: PEAUNITEDMASK - PCI Express* Unit Error Detect Mask Register

This register is specific to the IMCH, and controls detection of the PCI Express* functional unit error conditions. These bits are sticky through reset.

Table 16-216.Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 158h **Offset End:** 15Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 158h **Offset End:** 15Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0000h	
15	UPQODM	**Upstream Posted Queue Overflow Detect Mask.** This bit is sticky through reset. 0 = Enable upstream posted queue overflow detection. 1 = Disable upstream posted queue overflow detection.	Y	0b	RW
14	UNPQODM	**Upstream Non-Posted Queue Overflow Detect Mask.** This bit is sticky through reset. 0 = Enable upstream non-posted queue overflow detection. 1 = Disable upstream non-posted queue overflow detection.	Y	0b	RW
13	UCQODM	**Upstream Completion Queue Overflow Detect Mask** This bit is sticky through reset. 0 = Enable completion queue overflow detection. 1 = Disable completion queue overflow detection.	Y	0b	RW
12	LLEPEDM	**LLE Protocol Error Detect Mask** This bit is sticky through reset. 0 = Enable LLE protocol error detection. 1 = Disable LLE protocol error detection.	Y	0b	RW
11	MDLT	**Mask detection of link transitions from DL_UP to DL_DOWN:** 0 = Enable link down error detection. 1 = Disable link down error detection.		0b	RW
10	DDQPEDM	**Downstream Data Queue Parity Error Detect Mask** This bit is sticky through reset. 0 = Enable downstream data queue parity error detection. 1 = Disable downstream data queue parity error detection.	Y	0b	RW
09	Reserved	Reserved		0b	
08	Reserved	Reserved		0b	
07	Reserved	Reserved		0b	
06	Reserved	Reserved		0b	
05	Reserved	Reserved.		0b	
04 : 03	Reserved	Reserved		0b	

Table 16-216.Offset 158h: PEAUNITEDMASK - PCI Express Unit Error Detect Mask Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 158h **Offset End:** 15Bh
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 158h **Offset End:** 15Bh
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	MSCLTEDM	**Mask SMB Clock Low Timeout Error Detect Mask** This bit is sticky through reset. 0 = Enable SMB Clock Low Timeout error detection. 1 = Disable SMB Clock Low Timeout error detection.	Y	0b	RW
01	MUNSEDM	**Mask Unexpected NAK on SMB Error Detect Mask** **This** bit is sticky through reset. 0 = Enable Unexpected NAK on SMB error detection. 1 = Disable Unexpected NAK on SMB error detection.	Y	0b	RW
00	MSLBAEDM	**Mask SMB lost Bus Arbitration Error Detect Mistakes** bit is sticky through reset. 0 = Enable SMB arbitration loss detection. 1 = Disable SMB arbitration loss detection.	Y	0b	RW

16.4.1.78 Offset 160h: PEAFERR - PCI Express* First Error Register

Locks after first error.

Table 16-217.Offset 160h: PEAFERR - PCI Express First Error Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 160h **Offset End:** 163h
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 160h **Offset End:** 163h
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 09	Reserved	Reserved		000000h	
08	DFED	**Device Fatal Error Detected [STICKY]:** This bit is for internally detected fatal errors. 0 = No error detected 1 = Error Detected	Y	0b	RWC
07	DNFED	**Device Non-Fatal Error Detected [STICKY]:** This bit is for internally detected non-fatal errors. 0 = No error detected 1 = Error Detected	Y	0b	RWC
06	DCED	**Device Correctable Error Detected [STICKY]:** This bit is for internally detected correctable errors. 0 = No error detected 1 = Error Detected	Y	0b	RWC

Table 16-217.Offset 160h: PEAFERR - PCI Express First Error Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 160h **Offset End:** 163h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 160h **Offset End:** 163h	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
05	USFED	**Unit Specific Fatal Error Detected [STICKY]:** This bit is for fatal errors not in the PCI Express* specification as logged by the PEAUNITERR register. The PEAMASKERR register only prevents reporting of the unit errors, but does not prevent the logging of errors in this register. 0 = No error detected 1 = Error Detected		Y	0b	RWC
04	USNFED	**Unit Specific Non-Fatal Error Detected [STICKY]:** This bit is for non-fatal errors not in the PCI Express* specification as logged by the PEAUNITERR register. The PEAMASKERR register only prevents reporting of the unit errors, but does not prevent the logging of errors in this register. 0 = No error detected 1 = Error Detected		Y	0b	RWC
03	USCED	**Unit Specific Correctable Error Detected [STICKY]:** This bit is for correctable errors not in the PCI Express* specification as logged by the PEAUNITERR register. The PEAMASKERR register only prevents reporting of the unit errors, but does not prevent the logging of errors in this register. 0 = No error detected 1 = Error Detected		Y	0b	RWC
02	FEMR	**Fatal Error Message Received [STICKY]:** This bit is not set for internally detected fatal errors a.k.a. virtual fatal messages. These received fatal error messages can be masked by the SERR enable bit in the Bridge Control Register, if the SERR enable bit is a 0. 0 = No ERR_FATAL message received 1 = An ERR_FATAL message is received.		Y	0b	RWC
01	NFEMR	**Non-fatal Error Message Received [STICKY]:** This bit is not set for internally detected non-fatal errors a.k.a. virtual non-fatal messages. These received non-fatal error messages can be masked by the SERR enable bit in the Bridge Control Register, if the SERR enable bit is a 0. 0 = No ERR_NONFATAL message received 1 = An ERR_NONFATAL message is received.		Y	0b	RWC
00	CEMR	**Correctable Error Message Received [STICKY]:** This bit is not set for internally detected correctable errors a.k.a. virtual correctable messages. These received correctable error messages can be masked by the SERR enable bit in the Bridge Control Register, if the SERR enable bit is a 0. 0 = No ERR_COR message received 1 = An ERR_COR message is received.		Y	0b	RWC

16.4.1.79 Offset 164h: PEANERR - PCI Express* Next Error Register

Logs errors after FERR register is locks.

Table 16-218.Offset 164h: PEANERR - PCI Express Next Error Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 164h **Offset End:** 167h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 164h **Offset End:** 167h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
See Section 16.4.1.78, "Offset 160h: PEAFERR - PCI Express* First Error Register" for bit definitions.					

16.4.1.80 Offset 168h: PEAERRINJCTL - Error Injection Control Register

This register enables the injection of errors on incoming data streams into the core. The lower 16 bits are the corresponding flip parity bits for the cacheline of data. The upper bits in the register are for the use and control of the associated flip parity bits.

Table 16-219.Offset 168h: PEAERRINJCTL - Error Injection Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0	**Offset Start:** 168h **Offset End:** 16Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0	**Offset Start:** 168h **Offset End:** 16Bh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		00h	
19	SSB	**Stop and Scream Bit:** This is a special control for errors going to PCI Express*, outgoing from the core. Otherwise outgoing data errors are propagated. Not Supported for PCI Express*. 0 = Data errors are propagated. 1 = Data errors are not propagated only reported		0b	RO
18	EDDP	**Enable / Disable Data Poisoning:** 0 = Disable data poisoning - Errors won't be propagated, only good parity is generated. 1 = Enable data poisoning. Error Injection is possible regardless of this bit setting.		0b	RW

Table 16-219.Offset 168h: PEAERRINJCTL - Error Injection Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** 0:2:0		**Offset Start:** 168h **Offset End:** 16Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** 0:3:0		**Offset Start:** 168h **Offset End:** 16Bh	
Size: 32 bit	**Default:**	00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
17	FTDP	**Flip the designated parity bits:** 0 = No flip 1 = Flip the designated parity bits (bits 15:00) on all data transfers into the core. If a cacheline is in progress when this register is written, wait until the start of the next cacheline to flip parity bits.			0b	RW
16	FTPBNDT	**Flip the parity bits on just the next data transfer:** 0 = No flip 1 = Flip the designated parity bits (bits 15:00) in only the next data transfer into the core. If a cacheline is in progress when this register is written, wait until the start of the next cacheline to flip parity bits, in order to ensure all bits flipped are within the same cacheline. Hardware clears this when the injection has been performed. It is possible that the error injection desired did not occur because the next data transfer was not a complete cacheline, and the error to inject was in a different portion of the CL than was transferred. The hardware still clears the inject error once bit in this case. For completions which do not have the complete address, they are assumed to be 16B aligned addresses and only use implied address bits 03:02 to steer the parity error injection to the appropriate DW. ***Note:*** Since read completions are a maximum of 32B, half of the injection bits are not utilized.			0b	RWS
15 : 00	PI	**Parity Inject Bits:** Two bits of parity for each 64 B of data, 16 bits of parity for a cacheline.			0000h	RW

16.5 Memory Mapped I/O Registers for DRAM Controller

Table 16-220.Bus 0, Device 0, Function 0: Summary of IMCH SMRBASE Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: NOTESPAD - Note (Sticky) Pad for BIOS Support Register" on page 601	0000h
02h	03h	"Offset 02h: NOTEPAD - Note Pad for BIOS Support Register" on page 601	0000h
40h	43h	"Offset 40h: DCALCSR – DCAL Control and Status Register" on page 602	00000000h
44h	47h	"Offset 44h: DCALADDR - DCAL Address Register" on page 606	00000000h
48h at 1h	48h at 1h	"Offset 48h: DCALDATA[0-71] - DRAM Calibration Data Register" on page 607	00000000h
94h	96h	"Offset 94h: RCVENAC - Receiver Enable Algorithm Control Register" on page 611	180810h
98h	9Bh	"Offset 98h: DSRETC - DRAM Self-Refresh (SR) Extended Timing and Control Register" on page 611	5c141400h
9Ch	9Ch	"Offset 9Ch: DQSFAIL1 - DQS Failure Configuration Register 1" on page 612	00h
A0h	A3h	"Offset A0h: DQSFAIL0 - DQS Failure Configuration Register 0" on page 613	00000000h
A4h	A7h	"Offset A4h: DRRTC00 - Receive Enable Reference Output Timing Control Register" on page 615	06060606h
A8h	ABh	"Offset A8h: DRRTC01 - Receive Enable Reference Output Timing Control Register" on page 616	06060606h
C4h	C4h	"Offset C4h: DRRTC02 - Receive Enable Reference Output Timing Control Register" on page 616	06h
B4h	B7h	"Offset B4h: DQSOFCS00 - DQS Calibration Register" on page 617	00000000h
B8h	BBh	"Offset B8h: DQSOFCS01 - DQS Calibration Register" on page 617	00000000h
C6h	C6h	"Offset C6h: DQSOFCS02 - DQS Calibration Register" on page 618	00h
BCh	BFh	"Offset BCh: DQSOFCS10 - DQS Calibration Register" on page 618	00000000h
C0h	C3h	"Offset C0h: DQSOFCS11 - DQS Calibration Register" on page 619	00000000h
C7h	C7h	"Offset C7h: DQSOFCS12 - DQS Calibration Register" on page 619	00h
CCh	CFh	"Offset CCh: WPTRTC0 - Write Pointer Timing Control Register" on page 620	00000000h
D0h	D0h	"Offset D0h: WPTRTC1 - Write Pointer Timing Control 1 Register" on page 621	00h
D4h	D7h	"Offset D4h: DDQSCVDP0 - DQS Delay Calibration Victim Pattern 0 Register" on page 621	aaaa0a05h
D8h	DBh	"Offset D8h: DDQSCVDP1 - DQS Delay Calibration Victim Pattern 1 Register" on page 622	5b339c5dh
DCh	DFh	"Offset DCh: DDQSCADP0 - DQS Delay Calibration Aggressor Pattern 0 Register" on page 622	aaabffffh
E0h	E3h	"Offset E0h: DDQSCADP1 - DQS Delay Calibration Aggressor Pattern 1 Register" on page 623	db339ce1h
F0h	F3h	"Offset F0h: DIOMON - DDR I/O Monitor Register" on page 623	00000000h
F8h	FBh	"Offset F8h: DRAMISCTL - Miscellaneous DRAM DDR Cluster Control Register" on page 624	1011h
C8h	CAh	"Offset C8h: DRAMDLLC - DDR I/O DLL Control Register" on page 625	0DB6C0h
E8h	EBh	"Offset E8h: FIVESREG - Fixed 5s Pattern Register" on page 625	55555555h
ECh	EFh	"Offset ECh: AAAAREG - Fixed A Pattern Register" on page 626	AAAAAAAAh
140h	143h	"Offset 140h: MBCSR - MemBIST Control Register" on page 626	00000000h
144h	147h	"Offset 144h: MBADDR - Memory Test Address Register" on page 629	00h

Table 16-220.Bus 0, Device 0, Function 0: Summary of IMCH SMRBASE Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
148h at 4h	14Ch at 4h	"Offset 148h: MBDATA[0:9] - Memory Test Data Register" on page 629	00h
19Ch	19Fh	"Offset 19Ch: MB_START_ADDR - Memory Test Start Address Register" on page 632	00h
1A0h	1A3h	"Offset 1A0h: MB_END_ADDR - Memory Test End Address Register" on page 632	00h
1A4h	1A7h	"Offset 1A4h: MBLFSRSED - Memory Test Circular Shift and LFSR Seed Register" on page 633	00h
1A8h	1ABh	"Offset 1A8h: MBFADDRPTR - Memory Test Failure Address Pointer Register" on page 633	00h
1B0h	1B3h	"Offset 1B0h: MB_ERR_DATA00 - Memory Test Error Data 0" on page 634	00h
1B4h	1B7h	"Offset 1B4h: MB_ERR_DATA01 - Memory Test Error Data 0" on page 634	00h
1B8h	1BBh	"Offset 1B8h: MB_ERR_DATA02 - Memory Test Error Data 0" on page 634	00h
1BCh	1BFh	"Offset 1BCh: MB_ERR_DATA03 - Memory Test Error Data 0" on page 635	00h
1C0h	1C1h	"Offset 1C0h: MB_ERR_DATA04 - Memory Test Error Data 0" on page 635	00h
1C4h	1C7h	"Offset 1C4h: MB_ERR_DATA10 - Memory Test Error Data 1" on page 635	00h
1C8h	1CBh	"Offset 1C8h: MB_ERR_DATA11 - Memory Test Error Data 1" on page 636	00h
1CCh	1CFh	"Offset 1CCh: MB_ERR_DATA12 - Memory Test Error Data 1" on page 636	00h
1D0h	1D3h	"Offset 1D0h: MB_ERR_DATA13 - Memory Test Error Data 1" on page 636	00h
1D4h	1D5h	"Offset 1D4h: MB_ERR_DATA14 - Memory Test Error Data 1" on page 637	00h
1D8h	1DBh	"Offset 1D8h: MB_ERR_DATA20 - Memory Test Error Data 2" on page 637	00h
1DCh	1DFh	"Offset 1DCh: MB_ERR_DATA21 - Memory Test Error Data 2" on page 637	00h
1E0h	1E3h	"Offset 1E0h: MB_ERR_DATA22 - Memory Test Error Data 2" on page 638	00h
1E4h	1E7h	"Offset 1E4h: MB_ERR_DATA23 - Memory Test Error Data 2" on page 638	00h
1E8h	1E9h	"Offset 1E8h: MB_ERR_DATA24 - Memory Test Error Data 2" on page 638	00h
1ECh	1EFh	"Offset 1ECh: MB_ERR_DATA30 - Memory Test Error Data 3" on page 639	00h
1F0h	1F4h	"Offset 1F0h: MB_ERR_DATA31 - Memory Test Error Data 3" on page 639	00h
1F4h	1F7h	"Offset 1F4h: MB_ERR_DATA32 - Memory Test Error Data 3" on page 639	00h
1F8h	1FBh	"Offset 1F8h: MB_ERR_DATA33 - Memory Test Error Data 3" on page 640	00h
1FCh	1FDh	"Offset 1FCh: MB_ERR_DATA34 - Memory Test Error Data 3" on page 640	00h
260h	263h	"Offset 260h: DDRIOMC0 - DDRIO Mode Register Control Register" on page 641	00000078h
264h	267h	"Offset 264h: DDRIOMC1 - DDRIO Mode Register Control Register 1" on page 642	52520000h
268h	26Bh	"Offset 268h: DDRIOMC2 - DDRIO Mode Control Register 2" on page 645	039E6000h
284h at 4h	294h at 4h	"Offset 284h: WL_CNTL[4:0] - Write Levelization Control Register" on page 647	00000000h
298h	29Bh	"Offset 298h: WDLL_MISC - DLL Miscellaneous Control" on page 649	00000000h

16.5.1 Detailed Register Description

16.5.1.1 Offset 00h: NOTESPAD - Note (Sticky) Pad for BIOS Support Register

This dedicated 16-bit register is provided for BIOS.

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

Table 16-221.Offset 00h: NOTESPAD - Note (Sticky) Pad for BIOS Support Register

Description:					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	BSR	**BIOS Sticky Register [STICKY]:** This register is used by BIOS. It is sticky through reset.	Y	0000h	RW

16.5.1.2 Offset 02h: NOTEPAD - Note Pad for BIOS Support Register

This dedicated 16 bit register is provided for BIOS.

Table 16-222.Offset 02h: NOTEPAD - Note Pad for BIOS Support Register

Description:					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	BNSR	**BIOS Register:** This register is used by BIOS.	N	0000h	RW

16.5.1.3 Offset 40h: DCALCSR – DDR Calibration Control and Status Register

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

Note: DCALCSR is used only for calibration. MBCSR is used for Memory Test.

Table 16-223.Offset 40h: DCALCSR – DCAL Control and Status Register (Sheet 1 of 2)

Description: DCALCSR - DCAL Control and Status Register					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	START	**Start Operation** When set to 1 by software, the operation selected by the DCALCSR.OPCODE is initiated. Hardware clears this bit when the operation is complete.	N	0b	RWS
30 :28	FAIL	**Completion Status** 1xx = Fail, 0xx = Pass Note: Best practice is to rely on MemBIST following calibration to confirm a reliable DRAM interface.	N	000b	RW
27	BASPAT	**Basic Data Pattern Enable:** This controls which data pattern is used for the DQS Delay calibration. Setting this field enables the use of the basic data pattern selected by the DCALCSR.PATTERN bits. When cleared, the extended data pattern specified in the DDQSCVDP and DDQSCADP registers is used. Note: extended data pattern mode is not to be used in 2T configurations.	N	0b	RW
26	RSTREGSS	**Reset Registers in Single Step Mode:** Reset DCALDATA CSR in single step calibration mode. This bit should be set during the first step of a single step calibration. It will enable hardware to clear all registers and status bits during the calibration step the same way hardware does on the first step of an automatic "all passes" calibration.	N	0b	RW
25 :24	Reserved	Reserved	N	00b	RO
23	SGLSTP	**Single Step Calibration Operation:** Applies only to Receive enable and DQS cal. "1" = Single step - a single step of the algorithm selected by the DCALCSR.OPCODE is run by hardware. No data analysis is run. "0" = All passes - all steps of the algorithm selected by the DACLCSR.OPCODE is run by hardware including data analysis.	N	0b	RW
22 :21	CS	**Chip select:** This field corresponds to the chip select outputs: CS[1:0]. This field Applies to NOP, Refresh, Precharge all, and MRS/EMRS commands. It also applies to Receive Enable, and DQS Delay cal in single step mode. 01: select Rank 0 10: select Rank 1 00: Reserved 11: Reserved ***Note:*** Set CS to 01 for Self Refresh Entry. Hardware will automatically detect presence of a second rank/DIMM and sequence Self Refresh Entry via both chip selects if necessary.	N	00b	RW

Table 16-223.Offset 40h: DCALCSR – DCAL Control and Status Register (Sheet 2 of 2)

Description: DCALCSR - DCAL Control and Status Register					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
20 :19	Reserved	Reserved	N	00b	RO
18 :16	PAT	**Data pattern:** for DQS cal. This sets the burst length 4 pattern for a nibble of data. The pattern is repeated for BL8. This pattern is replicated on all nibbles of the data bus. “000” = F > 0 > F > 0 “001” = 0 > F > 0 > F “010” = A > 5 > A > 5 “011” = 5 > A > 5 > A “100” = C > 3 > C > 3 “101” = 3 > C > 3 > C “110” = 9 > 6 > 9 > 6 “111” = 6 > 9 > 6 > 9	N	000b	RW
15	DARWPR	**Disable FIFO reset:** in single pass mode. Applies only to Receiver enable and DQS cal. When set to 1, this bit inhibits the core to DDR cluster reset signal generated during the calibration modes. This prevents the DDR cluster synchronizer FIFO write pointer and data latches from being reset so that they can be read out of the cluster using the error monitor function. The reset signal can only be disabled in single step mode. When the DCALCSR.SGLSTP bit is set to 0, the DARWPR bit has no effect.	N	0b	RW
14 :04	OPMODS	**Operation modifiers:** See Table 16-224, Table 16-224, Table 16-227, and Table 16-235 for details	N	000h	RW
3 :00	OPCODE	**OPCODE:** “0000” = NOP “0001” = Refresh (SeeTable 16-226) “0010” = Pre-Charge “0011” = MRS/EMRS “0100” = Self-Refresh-Exit (SeeTable 16-226) “0101” = Automatic DQS Delay Calibration “0110”= Reserved “0111” = DLL BIST “1100” = Automatic Receive Enable Calibration “1101” = Self-Refresh Entry (SeeTable 16-226) “1110” = Error Monitor/Read DDRIO FIFO “1111” = ZQ Calibration All other settings are reserved	N	0000b	RW

Table 16-224.DCALCSR.OPMODS in Receive Enable Mode

Bit	Description
14:7	DRRTC override value to use in single step mode
6:4	Test point repeat number: The number of times each receive enable delay value is tested in order to reduce the effects of noise when near a timing threshold. When set to zero, hardware will repeat each step 8 times, the maximum number possible.

Table 16-225.DCALCSR.OPMODS in ZQ Calibration Mode

Bit	Description
14:3	Ignored
4	=0 ZQ Calibration Long =1 ZQ Calibration Short

Table 16-226.Rules about issuing Self-Refresh and Refresh commands using DCALCSR.OPCODE

1	The hardware does not enforce blocking of commands for tRFC period when a REFRESH cycle is launched using the DCALCSR.OPCODE. SW is responsible for ensuring that the refresh cycle time requirement is met.
2	Hardware will update DRC.CKE[1:0] bits with a self-refresh (SR) entry or exit command is issued using DCALCSR.OPCODE.
3	When issuing a self-refresh entry command using DCALCSR.OPCODE, the DCALCSR.CS needs to be set appropriately. The self-refresh entry commands will be issued on a per rank basis.
4	When issuing a self-refresh exit command using DCALCSR.OPCODE, the DCALCSR.CS will be ignored. Hardware will issue the second SR exit command if a second rank is present. Also note that a a self-refresh exit command using the DCALCSR.OPCODE can be issued only once per reset cycle. Hardware does support an additional mechanism through which a self-refresh command can be issued via DRC.CKE[1:0] (please refer to Section 11.4.5, “Self-Refresh” on page 302 for more details)

Table 16-227.DCALCSR.OPMODS in DQS Cal Mode

Bit	Description
14:12	Single step fine DLL delay: This is equivalent to DRAMDLLC.SLVLEN[4:0] when running DQS cal in single step mode
11:8	Single step coarse DLL delay: This is equivalent to DQSOFCS when running DQS cal in single step mode.
6:4	Test point repeat number: The number of times each delay value is tested in order to reduce the effects of noise when near a timing threshold. When the DCALCSR.BASPAT bit is set to select the basic data pattern, this field sets the number of times to repeat, with zero setting a max repeat value of 8. When the extended data pattern is selected, the max repeat value becomes 15. The max repeat is still selected with a value of zero, and other values result in a number of repeats equal to 2x-1.

Table 16-228.DCALCSR.OPMODS in Error Monitor/Read DDRIO FIFO Mode

Bit	Description
14:3	Reserved
2:0	DDRIO FIFO Entry number. 000: Entry 0 001: Entry 1 010: Entry 2 011: Entry 3 100: Entry 4 Others: Reserved

16.5.1.4 Offset 44h: DCALADDR - DDR Calibration Address Register

Table 16-229.Offset 44h: DCALADDR - DCAL Address Register

Description:	DCALADDR - DCAL Address Register				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 44h **Offset End:** 47h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DCALADDR	DCAL Address and Other Information based on DCALCSR.OPCODE. See Table 16-230.	N	00000000h	RW

Table 16-230.Interpretation of DCALADDR (Sheet 1 of 2)based on DCALCSR.OPCODE

Bit	NOP, Refresh, Pre-Charge, MRS/EMRS, and Self-Refresh Entry Commands initiated by DCALCSR	Receive Enable	DQS Delay Cal
31	DRAM Address Bus 15:0	Row Address 15:0	Row Address 15:0
30			
29			
28			
27			
26			
25			
24			
23			
22			
21			
20			
19			
18			
17			
16			

Table 16-230.Interpretation of DCALADDR (Sheet 2 of 2)based on DCALCSR.OPCODE

Bit	NOP, Refresh, Pre-Charge, MRS/EMRS, and Self-Refresh Entry Commands initiated by DCALCSR	Receive Enable	DQS Delay Cal
15		Column Address 15 to 11 & 9 to 2	Column Address 15 to 11 & 9 to 2
14			
13			
12			
11			
10			
9			
8			
7			
6			
5			
4			
3			
2	DRAM Bank Address Bus 2:0	Bank Address 2:0	Bank Address 2:0 in DCALCSR.BASPAT=1 mode, not used in DCALCSR.BASPAT=0 mode
1			
0			

16.5.1.5 Offset 48h: DCALDATA[0-71] - DRAM Calibration Data Registers

Table 16-231.Offset 48h: DCALDATA[0-71] - DRAM Calibration Data Register

Description:	DCALData - DRAM Calibration Data Registers				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 48h at 1h **Offset End:** 48h at 1h	
Size: 8 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	DCALDATA	DCAL Data and other information based on DCALCSR.OPCODE. See Table 16-232.	N	00h	RW

Table 16-232.DCALDATA Based on DCALCSR.OPCODE

Byte	Receive Enable	DQS Cal	Error Monitor/Read DDRIO FIFO
71			Not Used
70			
69	Preamble status DQS8 rank1	Max delay DQS8 rank1	
68	First edge position DQS8 rank1	Min delay DQS8 rank1	
67			
66			
65	Preamble status DQS7 rank1	Max delay DQS7 rank1	
64	First edge position DQS7 rank1	Min delay DQS7 rank1	
63			
62			
61	Preamble status DQS6 rank1	Max delay DQS6 rank1	
60	First edge position DQS6 rank1	Min delay DQS6 rank1	
59			
58			
57	Preamble status DQS5 rank1	Max delay DQS5 rank1	
56	First edge position DQS5 rank1	Min delay DQS5 rank1	
55			
54			
53	Preamble status DQS4 rank1	Max delay DQS4 rank1	
52	First edge position DQS4 rank1	Min delay DQS4 rank1	
51			
50			
49	Preamble status DQS3 rank1	Max delay DQS3 rank1	
48	First edge position DQS3 rank1	Min delay DQS3 rank1	
47			
46			
45	Preamble status DQS2 rank1	Max delay DQS2 rank1	
44	First edge position DQS2 rank1	Min delay DQS2 rank1	
43			
42			
41	Preamble status DQS1 rank1	Max delay DQS1 rank1	
40	First edge position DQS1 rank1	Min delay DQS1 rank1	
39			
38			
37	Preamble status DQS0 rank1	Max delay DQS0 rank1	
36	First edge position DQS0 rank1	Min delay DQS0 rank1	

Byte	Receive Enable	DQS Cal	Error Monitor/Read DDRIO FIFO
35			Even or early 72 bits of read data from the DDRIO FIFO
34			
33	Preamble status DQS8 rank0	Max delay DQS8 rank0	
32	First edge position DQS8 rank0	Min delay DQS8 rank0	
31			
30			
29	Preamble status DQS7 rank0	Max delay DQS7 rank0	
28	First edge position DQS7 rank0	Min delay DQS7 rank0	
27			
26			Odd or late 72 bits of read data from the DDRIO FIFO
25	Preamble status DQS6 rank0	Max delay DQS6 rank0	
24	First edge position DQS6 rank0	Min delay DQS6 rank0	
23			
22			
21	Preamble status DQS5 rank0	Max delay DQS5 rank0	
20	First edge position DQS5 rank0	Min delay DQS5 rank0	
19			
18			
17	Preamble status DQS4 rank0	Max delay DQS4 rank0	Not Used
16	First edge position DQS4 rank0	Min delay DQS4 rank0	
15			
14			
13	Preamble status DQS3 rank0	Max delay DQS3 rank0	
12	First edge position DQS3 rank0	Min delay DQS3 rank0	
11			
10			
9	Preamble status DQS2 rank0	Max delay DQS2 rank0	
8	First edge position DQS2 rank0	Min delay DQS2 rank0	
7			
6			
5	Preamble status DQS1 rank0	Max delay DQS1 rank0	
4	First edge position DQS1 rank0	Min delay DQS1 rank0	
3			
2			
1	Preamble status DQS0 rank0	Max delay DQS0 rank0	
0	First edge position DQS0 rank0	Min delay DQS0 rank0	

DCALDATA Receiver Enable “First edge position” byte detail	
Bit	**Description**
7:0	At the end of a successful calibration, this register holds the DRRTC setting that enables the DQS receiver as close as possible to but no earlier than the first rising DQS transition after the preamble. At the start of the calibration, this register is loaded with a value of 0xFF. During the calibration, while the “strobe toggle status” bit is low, this register will be updated with the DRRTC value for the current calibration step if the DQS is found to have a value of zero. After “strobe toggle status” goes high, this register will be updated with the DRRTC value when the DQS is found to have a value of one at a calibration step. This register will no longer be updated after the “preamble found status” bit goes high, so that it will retain the position of the rising DQS edge following immediately after the preamble.

DCALDATA Receiver Enable “Preamble status” byte detail	
Bit	**Description**
7	Strobe toggle status. Hardware sets this bit if a valid high pulse is found in the strobe waveform. The requirement is (DCALDATA.First_edge_position - last receiver enable delay value) > RCVENAC.HWIDTH
6	Preamble found status. Hardware sets this bit if the “preamble found” bit asserts at any time during the calibration.
5	Preamble found. Last receiver enable delay value meets or exceeds the preamble width requirement setting. Hardware sets this bit if: (DCALDATA.First_edge_position - last receiver enable delay value) > RCVENAC.PWIDTH
4:0	Count of “lows” minus count of “highs” found during one set of repeated tests at the last receiver enable delay setting. See DCALCSR opmods field for a description of the repeat test function.

DCALDATA DQS Cal Min Delay detail	
Bit	**Description**
7:6	reserved
5:0	At the end of a successful calibration, this field will hold the minimum DQS delay setting that results in correct data capture in the DDR I/O capture flop. This is the left edge of the DQ data eye. During the calibration, this field is updated with the DQS delay setting of the current calibration step, until correct data capture is found. After this point no further updates are made.

DCALDATA DQS Cal Max Delay detail	
Bit	**Description**
7:6	reserved
5:0	At the end of a successful calibration, this field will hold the maximum DQS delay setting that results in correct data capture in the DDR I/O capture flop. This is the right edge of the DQ data eye. During the calibration, this field is updated with the DQS delay setting at each calibration step until the minimum delay setting is found and a subsequent failure to capture correct read data occurs.

16.5.1.6 Offset 94h: RCVENAC - Receiver Enable Algorithm Control Register

This register contains controls for the preamble detection algorithm of the automatic receiver enable logic. RCVENAC.PWIDTH is used to determine if a “low” pulse in a DQS waveform is wide enough to be a preamble. RCVENAC.POFFSET is subtracted from the DCALDATA first edge position result and programmed into the DRRTC registers

Table 16-233.Offset 94h: RCVENAC - Receiver Enable Algorithm Control Register

Description:	RCVENAC: Receiver Enable Algorithm Control					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0		**Offset Start:** 94h **Offset End:** 96h	
Size: 24 bit	**Default:** 180810h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
23 :16	PWIDTH	**Minimum preamble width limit**, used to detect if a low pulse in a DQS waveform is wide enough to be a valid preamble. The default corresponds to 3/4 of a DRAM clock cycle		Y	18h	RW
15 :14	Reserved	Reserved			00b	RO
13 :08	HWIDTH	**Minimum high pulse width limi**t, used to detect if a high pulse in a DQS waveform is wide enough to indicate a strobe is toggling in a valid manner. The default corresponds to 1/4 of a DRAM clock cycle.		Y	08h	RW
7 :06	Reserved	Reserved			00b	RO
5 :00	POFFSET	**Preamble center offset** from first rising edge, used to position the DQS receiver enable relative to the preamble edge location recorded in the DCALDATA registers. The default value corresponds to 1/2 of a DRAM clock cycle.		Y	10h	RW

16.5.1.7 Offset 98h: DSRETC - DRAM Self-Refresh (SR) Extended Timing and Control

This register implements bits fields that control self-refresh entry and exit mechanisms that are required for ACPI S3 mode of operation.

Table 16-234.Offset 98h: DSRETC - DRAM Self-Refresh (SR) Extended Timing and Control Register

Description:	DSRETC: DRAM Self-Refresh (SR) Extended Timing and Control Register					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0		**Offset Start:** 98h **Offset End:** 9Bh	
Size: 32 bit	**Default:** 5c141400h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	TXSNR	Exit self-refresh to non-read command timing. Number of Controller cycles for which accesses to the DIMMs need to be blocked by memory controller.		Y	01011100b	RW
23 :16	DRSRENT	Dual rank self-refresh (SR) entry and exit timing - stagger of self refresh commands between ranks. Staggering of the SR commands result is in the power intensive refresh operations to be staggered between the 2 ranks.		Y	00010100b	RW

Table 16-234.Offset 98h: DSRETC - DRAM Self-Refresh (SR) Extended Timing and Control Register

Description:	DSRETC: DRAM Self-Refresh (SR) Extended Timing and Control Register				
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0	Offset Start: 98h Offset End: 9Bh	
Size: 32 bit	Default: 5c141400h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	DRARTIM	Dual rank auto-refresh timing - stagger of commands between ranks prior to self-refresh entry.	Y	00010100b	RW
7 :1	Reserved	Reserved	N	0000000b	RO
0 0	ENSREXIT	Enable Self-refresh (SR) exit state machine. This bit needs to be set by BIOS upon power-up from an S3 event.	N	0b	RW

16.5.1.8 Offset 9Ch: DQSFAIL1 - DQSFAIL1 Configuration Register

There are two DQSFAIL registers that contain a total of 18 individual DQS failure status bits. There is one status bit for each DQS on each rank. These bits are set automatically by hardware during the receiver enable calibration if a valid DQS waveform is not detected. Hardware will not clear any bits that are set prior to the calibration even if a valid waveform is detected.

Hardware uses the DQSFAIL information to exclude calibration data during the data gathering portion and/or the data analysis portion of the both the receiver enable and DQS delay calibrations as well as MBIST. This prevents a failed DQS pin from corrupting the calibration of neighboring functional DQS pins that may share internal logic resources with a failing DQS pin.

For normal calibration, initialize all DQSFAIL bits to 0.

Table 16-235.Offset 9Ch: DQSFAIL1 - DQS Failure Configuration Register 1

Description:	DQSFAIL1: DQS Failure Configuration Register				
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0	Offset Start: 9Ch Offset End: 9Ch	
Size: 8 bit	Default: 00h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :04	Reserved	Reserved	N	0h	RO
03 03	Reserved_R1DQS17	Reserved	Y	0b	RW
02 02	R1DQS08	Rank 1 DQS08	Y	0b	RW
01 01	Reserved_R1DQS16	Reserved	Y	0b	RW
00 00	R1DQS07	Rank 1 DQS07	Y	0b	RW

16.5.1.9 Offset A0h: DQSFAIL0 - DQSFAIL0 Configuration Register

See description for DQSFAIL1 (Section 16.5.1.8).

Table 16-236.Offset A0h: DQSFAIL0 - DQS Failure Configuration Register 0

Description:	DQSFAIL0: DQS Failure Configuration Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved_R1DQS15	Reserved	Y	0b	RW
30	R1DQS06	Rank 1 DQS06	Y	0b	RW
29	Reserved_R1DQS14	Reserved	Y	0b	RW
28	R1DQS05	Rank 1 DQS05	Y	0b	RW
27	Reserved_R1DQS13	Reserved	Y	0b	RW
26	R1DQS04	Rank 1 DQS04	Y	0b	RW
25	Reserved_R1DQS12	Reserved	Y	0b	RW
24	R1DQS03	Rank 1 DQS03	Y	0b	RW
23	Reserved_R1DQS11	Reserved	Y	0b	RW
22	R1DQS02	Rank 1 DQS02	Y	0b	RW
21	Reserved_R1DQS10	Reserved	Y	0b	RW
20	R1DQS01	Rank 1 DQS01	Y	0b	RW
19	Reserved_R1DQS09	Reserved	Y	0b	RW
18	R1DQS00	Rank 1 DQS00	Y	0b	RW
17	Reserved_R0DQS17	Reserved	Y	0b	RW
16	R0DQS08	Rank 0 DQS08	Y	0b	RW
15	Reserved_R0DQS16	Reserved	Y	0b	RW
14	R0DQS07	Rank 0 DQS07	Y	0b	RW
13	Reserved_R0DQS15	Reserved	Y	0b	RW
12	R0DQS06	Rank 0 DQS06	Y	0b	RW
11	Reserved_R0DQS14	Reserved	Y	0b	RW
10	R0DQS05	Rank 0 DQS05	Y	0b	RW
9	Reserved_R0DQS13	Reserved	Y	0b	RW
8	R0DQS04	Rank 0 DQS04	Y	0b	RW
7	Reserved_R0DQS12	Reserved	Y	0b	RW
6	R0DQS03	Rank 0 DQS03	Y	0b	RW

Table 16-236.Offset A0h: DQSFAIL0 - DQS Failure Configuration Register 0

Description:	DQSFAIL0: DQS Failure Configuration Register				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
5	Reserved_R0DQS11	Reserved	Y	0b	RW
4	R0DQS02	Rank 0 DQS02	Y	0b	RW
3	Reserved_R0DQS10	Reserved	Y	0b	RW
2	R0DQS01	Rank 0 DQS01	Y	0b	RW
1	Reserved_R0DQS09	Reserved	Y	0b	RW
0	R0DQS00	Rank 0 DQS00	Y	0b	RW

16.5.1.10 DRRTC: Receive Enable Reference Output Timing Control Registers

Note: These registers have to be saved and restored on S3.

The DRRTC is a set of three registers with DQS receiver enable window timing control for each byte on the DDR data bus. There is a single control for each byte for both ranks. A correct register setting will delay the start of the enable window so that it coincides with the middle of the DQS pre-amble. Enabling the window before or after the pre-amble would cause valid DQS edges to be missed or invalid edges or noise to be received.

The range of the enable delay, controlled by the DRRTC registers, is eight cycles, with a granularity defined by the DDRIOMC2.MASTCNTL register (controls setting for the Master DLL). The delay is measured from the memory controller clock edge that launches a “read” command on the DDR command bus. The minimum delay is equal to the DDR SDRAM read latency defined in the DRT0.CL register field. The maximum delay is the read latency plus eight cycles. In order words the DRRTC registers can introduce up to 8 cycles of delay. This DRRTC delay does not included the contributions of CL and registered DIMM to the total read latency. In addition to these major sources of delay, there is also a small “uncompensated delay” as shown in the formulas below.

The RCVEN fields of the DRRTC register control the delay as follows: bits [7:5] control whole clock increments, bits [4:3] control in quarter clock increments, and bits [2:0] control the sub-quarter cycle increments. Setting RCVEN to 0x0 produces the minimum delay, and 0xFF sets the maximum delay. The sub-quarter cycle delay is defined by the equations and “RCVEN_OUT” lookup table below:

Delay_Uncomp = 100ps; Note: estimate only

Delay Element = (quarter CMDCLK period - Delay_Uncomp) / (MASTCNTL + 0.5)

sub quarter cycle delay = Delay_Uncomp + (Delay Element * RCVEN_OUT[2:0])

RCVEN_OUT Lookup Table								
DDRIOMC2 [MASTCNTL]	DRRTC RCVEN [2:0]							
	7	6	5	4	3	2	1	0
7	7	6	5	4	3	2	1	0
6	6	5	5	4	3	2	1	0
5	5	4	4	3	2	1	1	0
4	4	4	3	3	2	1	1	0
3	3	3	2	2	1	1	0	0
2	2	2	2	1	1	0	0	0
1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0

For example, if the DDRIOMC2.MASTCNTL is set to 0x7, the receiver enable delay can be varied over eight cycle in 256 steps, one step for each DRRTC RCVEN setting. If DDRIOMC2.MASTCNTL is set to 0x3, however, the number of steps is reduced to 128, such that half of the DRRTC RCVEN settings do not produce an increase in delay from the previous setting.

16.5.1.11 Offset A4h: DRRTC00 - Receive Enable Reference Output Timing Control Register

This register determines DQS 3, 2, 1, & 0 input buffer enable timing delay

Table 16-237.Offset A4h: DRRTC00 - Receive Enable Reference Output Timing Control Register

Description:	DRRTC00: Receive Enable Reference Output Timing Control Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** A4h **Offset End:** A7h	
Size: 32 bit	**Default:** 06060606h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RCVEN03	Receiver enable delay for DQS3	Y	06h	RW
23 :16	RCVEN02	Receiver enable delay for DQS2	Y	06h	RW
15 :08	RCVEN01	Receiver enable delay for DQS1	Y	06h	RW
7 :00	RCVEN00	Receiver enable delay for DQS0	Y	06h	RW

16.5.1.12 Offset A8h: DRRTC01 - Receive Enable Reference Output Timing Control Register

This register determines DQS 7, 6, 5, & 4 input buffer enable timing delay.

Table 16-238.Offset A8h: DRRTC01 - Receive Enable Reference Output Timing Control Register

Description:	DRRTC01: Receive Enable Reference Output Timing Control Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** A8h **Offset End:** ABh	
Size: 32 bit	**Default:** 06060606h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RCVEN07	Receiver enable delay for DQS7	Y	06h	RW
23 :16	RCVEN06	Receiver enable delay for DQS6	Y	06h	RW
15 :08	RCVEN05	Receiver enable delay for DQS5	Y	06h	RW
7 :00	RCVEN04	Receiver enable delay for DQS4	Y	06h	RW

16.5.1.13 Offset C4h: DRRTC02 - Receive Enable Reference Output Timing Control Register

This register determines DQS 8 input buffer enable timing delay.

Table 16-239.Offset C4h: DRRTC02 - Receive Enable Reference Output Timing Control Register

Description:	DRRTC02: Receive Enable Reference Output Timing Control Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** C4h **Offset End:** C4h	
Size: 8 bit	**Default:** 06h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7 :00	RCVEN08	Receiver enable delay for DQS8	Y	06h	RW

16.5.1.14 DQS Calibration Registers

The DQSOFCS is a group of six registers that control the fine delay used to center DQS edges to the DQ data eye during read operations. There is a delay entry for each nibble of the DDR data bus for each rank. The coarse delay is controlled by the DRAMDLLC register. The equations for the fine and coarse delays are shown below. Note that “Delay Element” and “Delay_Uncomp” are defined in the DRRTC register section. Also note that there is a separate coarse delay control for each “chunk” of the DDR I/O cluster as defined in the DRAMDLLC register section.

Note: these registers may have to be saved and restored on S3

16.5.1.15 Offset B4h: DQSOFCS00 - DQS Calibration Register

This register determines DQS3, 2, 1, & 0 fine DQS delay when reading from rank 0.

Table 16-240.Offset B4h: DQSOFCS00 - DQS Calibration Register

Description:	DQSOFCS00: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** B4h **Offset End:** B7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	Reserved	Reserved	N	0h	RO
27 :24	DQS03	Rank 0 DQS03: Fine delay	Y	0h	RW
23 :20	Reserved	Reserved	N	0h	RO
19 :16	DQS02	Rank 0 DQS02: Fine delay	Y	0h	RW
15 :12	Reserved	Reserved	N	0h	RO
11 :08	DQS01	Rank 0 DQS01: Fine delay	Y	0h	RW
07 :04	Reserved	Reserved	N	0h	RO
03 :00	DQS00	Rank 0 DQS00: Fine delay	Y	0h	RW

16.5.1.16 Offset B8h: DQSOFCS01 - DQS Calibration Register

This register determines DQS7, 6, 5, & 4 fine DQS delay when reading from rank 0.

Table 16-241.Offset B8h: DQSOFCS01 - DQS Calibration Register

Description:	DQSOFCS01: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** B8h **Offset End:** BBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	Reserved	Reserved	N	0h	RO
27 :24	DQS07	Rank 0 DQS07: Fine delay	Y	0h	RW
23 :20	Reserved	Reserved	N	0h	RO
19 :16	DQS06	Rank 0 DQS06: Fine delay	Y	0h	RW
15 :12	Reserved	Reserved	N	0h	RO
11 :08	DQS05	Rank 0 DQS05: Fine delay	Y	0h	RW
7 :04	Reserved	Reserved	N	0h	RO
3 :00	DQS04	Rank 0 DQS04: Fine delay	Y	0h	RW

16.5.1.17 Offset C6h: DQSOFCS02 - DQS Calibration Register

This register determines DQS 8 fine DQS delay when reading from rank 0.

Table 16-242.Offset C6h: DQSOFCS02 - DQS Calibration Register

Description:	DQSOFCS02: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** C6h **Offset End:** C6h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :04	Reserved	Reserved	N	0h	RO
03 :00	DQS08	Rank 0 DQS08: Fine delay	Y	0h	RW

16.5.1.18 Offset BCh: DQSOFCS10 - DQS Calibration Register

This register determines DQS 3, 2, 1, & 0 fine DQS delay when reading from rank 1.

Table 16-243.Offset BCh: DQSOFCS10 - DQS Calibration Register

Description:	DQSOFCS10: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** BCh **Offset End:** BFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	Reserved	Reserved	N	0h	RO
27 :24	DQS03	Rank 1 DQS03: Fine delay	Y	0h	RW
23 :20	Reserved	Reserved	N	0h	RO
19 :16	DQS02	Rank 1 DQS02: Fine delay	Y	0h	RW
15 :12	Reserved	Reserved	N	0h	RO
11 :08	DQS01	Rank 1 DQS01: Fine delay	Y	0h	RW
07 :04	Reserved	Reserved	N	0h	RO
03 :00	DQS00	Rank 1 DQS00: Fine delay	Y	0h	RW

16.5.1.19 Offset C0h: DQSOFCS11 - DQS Calibration Register

This register determines DQS7, 6, 5, & 4 fine DQS delay when reading from rank 1

Table 16-244.Offset C0h: DQSOFCS11 - DQS Calibration Register

Description:	DQSOFCS11: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** C0h **Offset End:** C3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	Reserved	Reserved	N	0h	RO
27 :24	DQS07	Rank 1 DQS07: Fine delay	Y	0h	RW
23 :20	Reserved	Reserved	N	0h	RO
19 :16	DQS06	Rank 1 DQS06: Fine delay	Y	0h	RW
15 :12	Reserved	Reserved	N	0h	RO
11 :08	DQS05	Rank 1 DQS05: Fine delay	Y	0h	RW
07 :04	Reserved	Reserved	N	0h	RO
03 :00	DQS04	Rank 1 DQS04: Fine delay	Y	0h	RW

16.5.1.20 Offset C7h: DQSOFCS12 - DQS Calibration Register

This register determines DQS 8 fine DQS delay when reading from rank 1

Table 16-245.Offset C7h: DQSOFCS12 - DQS Calibration Register

Description:	DQSOFCS12: DQS Calibration Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** C7h **Offset End:** C7h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :04	Reserved	Reserved	N	0h	RO
03 :00	DQS08	Rank 1 DQS08: Fine delay	Y	0h	RW

16.5.1.21 WPTRTC DDR I/O Write Pointer Timing

The two WPTRTC registers control the fine delay of the DDR I/O FIFO write pointers. The formulas for delay shown in the DQSOFCS and DRRTC register sections are identical to the write pointer delay formulas. To find the WPTRTC portion of write pointer delay, use the DQSOFCS formulas, and substitute WPTRTC fields for all the DQSOFCS fields. The only difference in the application of these formulas is that there is only one WPTRTC field per byte of the DDR I/O, whereas the DQSOFCS has a field per byte per rank. The total write pointer delay, measured from the same reference point as the DQS receiver enable timing, is equal to the DQS receiver enable timing, including the quarter clock and sub-quarter clock delays, plus one full clock cycle, plus the coarse and fine DRAMDLLC.SLVLEN/WPTRTC delays calculated with the formulas from the DQSOFCS register section.

16.5.1.22 Offset CCh: WPTRTC0 - Write Pointer Timing Control 0 Register

This register determines the DDR I/O FIFO write pointer fine delay timing for all DQS signals except DQS8 when reading from rank 0 or rank 1.

Table 16-246.Offset CCh: WPTRTC0 - Write Pointer Timing Control Register

Description:	WPTRTC0: Write pointer timing control				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** CCh **Offset End:** CFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	DQS07	DQS7 write pointer fine delay	Y	0h	RW
27 :24	DQS06	DQS6 write pointer fine delay	Y	0h	RW
23 :20	DQS05	DQS5 write pointer fine delay	Y	0h	RW
19 :16	DQS04	DQS4 write pointer fine delay	Y	0h	RW
15 :12	DQS03	DQS3 write pointer fine delay	Y	0h	RW
11 :08	DQS02	DQS2 write pointer fine delay	Y	0h	RW
7 :04	DQS01	DQS1 write pointer fine delay	Y	0h	RW
3 :00	DQS00	DQS0 write pointer fine delay	Y	0h	RW

16.5.1.23 Offset D0h: WPTRTC1 - Write Pointer Timing Control 1 Register

This register determines the DDR I/O FIFO write pointer fine delay timing for DQS8 signals when reading from rank 0 or rank 1

Table 16-247.Offset D0h: WPTRTC1 - Write Pointer Timing Control 1 Register

Description:	WPTRTC1: Write pointer timing control				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** D0h **Offset End:** D0h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7 :04	Reserved	Reserved		0h	RO
3 :00	DQS08	DQS8 write pointer fine delay	Y	0h	RW

16.5.1.24 DDQSCVDP and DDQSCADP

This set of 4 registers defines two 64 bit long data patterns used in the DQS Delay Calibration. They are only used when DCALCSR.BASPAT is low. The 64 bit patterns cover a data burst that is 32 DRAM clock cycles long. The DDQSCVDP registers define the “victim” pattern, and the DDQSCADP defines the “aggressor” pattern. The victim pattern is applied to one bit of each byte of the DDR data bus for 32 clock cycles, and the aggressor pattern is applied to all other bits. The victim pattern is applied in turn to each bit of each byte, creating a complete data pattern that is 8*32 data cycles long.

16.5.1.25 Offset D4h: DDQSCVDP0 - DQS DELAY CALIBRATION VICTIM PATTERN 0 Register

This register defines the last 32 bits of the 64 bit long “victim” data pattern.

Table 16-248.Offset D4h: DDQSCVDP0 - DQS Delay Calibration Victim Pattern 0 Register

Description:	DDQSCVDP0: DQS Delay Cal Pattern				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** D4h **Offset End:** D7h	
Size: 32 bit	**Default:** aaaa0a05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	VP0	Victim pattern 0		aaaa0a05h	RW

16.5.1.26 Offset D8h: DDQSCVDP1 - DQS DELAY CALIBRATION VICTIM PATTERN 1 Register

This register defines the first 32 bits of the 64 bit long “victim” data pattern.

Table 16-249.Offset D8h: DDQSCVDP1 - DQS Delay Calibration Victim Pattern 1 Register

Description: DDQSCVDP1: DQS Delay Cal Pattern					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** D8h **Offset End:** DBh	
Size: 32 bit	**Default:** 5b339c5dh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	VP1	Victim pattern 1		5b339c5dh	RW

16.5.1.27 Offset DCh: DDQSCADP0 - DQS DELAY CALIBRATION AGGRESSOR PATTERN 0 Register

This register defines the last 32 bits of the 64 bit long “aggressor” data pattern.

Table 16-250.Offset DCh: DDQSCADP0 - DQS Delay Calibration Aggressor Pattern 0 Register

Description: DDQSCADP0: DQS Delay Cal Pattern					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** DCh **Offset End:** DFh	
Size: 32 bit	**Default:** aaabffffh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	AP0	Aggressor pattern 0		aaabffffh	RW

16.5.1.28 Offset E0h: DDQSCADP1 - DQS DELAY CALIBRATION AGGRESSOR PATTERN 1 Register

This register defines the first 32 bits of the 64 bit long “aggressor” data pattern.

Table 16-251.Offset E0h: DDQSCADP1 - DQS Delay Calibration Aggressor Pattern 1 Register

Description:	DDQSCADP1: DQS Delay Cal Pattern				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** E0h **Offset End:** E3h	
Size: 32 bit	**Default:** db339ce1h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	AP1	Aggressor pattern 1		db339ce1h	RW

16.5.1.29 Offset F0h: DIOMON - DDR I/O Monitor Register

This register monitors the legsel output of the DDR I/O and controls the A/D converter in DDR I/O used to monitor analog voltage levels.

Table 16-252.Offset F0h: DIOMON - DDR I/O Monitor Register

Description:	DIOMON: DDR I/O Monitor				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** F0h **Offset End:** F3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 25	Reserved	Reserved	N	00000000b	RO
24 24	DSAMP	Causes the analog to digital converter to sample the analog input selected by biasssel	Y	0b	RW
23 16	VRESULT	A/D converter output of DDR I/O	Y	00000000b	RO
15 15	ENABLE	Enable A/D converter for the DDR IO Bias logic. Also enables updates to the following fields of this CSR: VRESULT, DQLEGSELOUT, DIOPWR, CALEGSELOUT	N	0b	RW
14 :11	BIASSEL	A/D converter input selection	Y	0000b	RW
10 07	DQLEGSELOUT	DQ legsel output of DDR I/O. Sets the driver strength for DQ IO buffers.	Y	0000b	RO
06 06	DIOPWR	Nopwr = 0 if Vccddr is off OR in burnin mode. During normal operation it's set to 1.	Y	0b	RO
05 :04	Reserved	Reserved	N	00b	RO
03 :00	CALEGSELOUT	cmd/addr legsel output of DDR I/O Sets the driver strength for cmd/addr IO buffers.	Y	0000b	RO

16.5.1.30 Offset F8h: DRAMISCTL - Miscellaneous DRAM DDR Cluster Control Register

Table 16-253.Offset F8h: DRAMISCTL - Miscellaneous DRAM DDR Cluster Control Register

Description:	DRAMISCTL: Miscellaneous DRAM DDR Cluster Control Register				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:** 1011h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :13	Reserved	Reserved	N	0b	RO
12 12	Reserved	Reserved	Y	1b	RW
11 11	Reserved	Reserved	N	0b	RW
10 8	Reserved_RW	Reserved for future use. These bits are RW but SW should not change the default reset value of these bits.	N	000b	RW
7 :0	VREFSEL	Vref selection: Adjustable VREF voltage at receivers. The threshold voltage at receiver can be raised or lowered to allow the noise margin on the data from memory be skewed. Vref is estimated with the following equation Vref = (SQU * VCCDDR + (SQD – SQU) * 0.45) / (SQU + SQD) + VOFF where, SQU = SQRT(4*VREFSEL<7> + 2*VREFSEL<6> + VREFSEL<5> + 8*VREFSEL<4>) SQD = SQRT(4*VREFSEL<3> + 2*VREFSEL<2> + VREFSEL<1> + 8*VREFSEL<0>) VOFF = offset, varying for each chip, nominal value is 0 but can be up to +/- 0.1V Examples with VCCDDR=1.8V and VOFF=0 .VREFSEL. Vref (V) 00010001 0.9 00010011 0.887 00010101 0.875 00011001 0.855 11101001 0.840 11001001 0.823 10001001 0.779 00110001 0.913 01010001 0.925 10010001 0.945 10011110 0.960 10011100 0.977 10011000 1.021	Y	11h	RW

16.5.1.31 Offset C8h: DRAMDLLC - DDR I/O DLL Control Register

The formulas that show how the SLVLEN fields affect DQS delay timing are shown in the DQSOFCS register definition section. The SLVLEN fields are set by hardware during the DQS delay calibration. There are five SLVLEN fields, one for each two bytes of the DDR I/O DQ pins. The SLVBYP bit can be toggled to reset the master DLL's in the DDR I/O.

Table 16-254.Offset C8h: DRAMDLLC - DDR I/O DLL Control Register

Description:	DRAMDLLC: DDR I/O DLL Control				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** C8h **Offset End:** CAh	
Size: 24 bit	**Default:** 0DB6C0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 :22	Reserved	Reserved	N	00b	RO
21	SLVBYP	DQS delay bypass	Y	0b	RW
20 :18	SLVLEN4	dqs 8 coarse DQS delay	Y	011b	RW
17 :15	SLVLEN3	dqs 7 & 6 coarse DQS delay	Y	011b	RW
14 :12	SLVLEN2	dqs 5 & 4 coarse DQS delay	Y	011b	RW
11 :09	SLVLEN1	dqs 3 & 2 coarse DQS delay	Y	011b	RW
8 :06	SLVLEN0	dqs1 & 0 coarse DQS delay	Y	011b	RW
5 :00	Reserved	Reserved	N	000000b	RO

16.5.1.32 Offset E8h: FIVESREG - Fixed 5s Pattern Register

Constant value used for debug.

Table 16-255.Offset E8h: FIVESREG - Fixed 5s Pattern Register

Description:	FIVESREG: Fixed 5s Pattern				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** E8h **Offset End:** EBh	
Size: 32 bit	**Default:** 55555555h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	FIVES	Hardwired to 5s for read-return	N	55555555h	RO

16.5.1.33 Offset ECh: AAAAREG - Fixed As Pattern Register

Constant value used for debug.

Table 16-256.Offset ECh: AAAAREG - Fixed A Pattern Register

Description: AAAAREG: Fixed A Pattern					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** ECh **Offset End:** EFh	
Size: 32 bit	**Default:** AAAAAAAAh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	AAAA	Hardwired to As for read-return	N	AAAAAAAAh	RO

16.5.1.34 Memory BIST Registers

The following set of registers cover the MemBIST logic which is used for testing the external DDR devices, as well as the connections between the EP80579 and the DDR chips. The following must be complete prior to running MemBist:

- Memory controller internal CSRs are set to match external DDR configuration.
- JEDEC DDRII/III initialization sequence.
- EP80579 DDR pad (Receive Enable and DQS) calibration.

The CSR's following this CSR are the memory-mapped registers for the memory controller. The SMRBASE register described in Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

16.5.1.35 Offset 140h: MBCSR - MemBIST Control Register

Table 16-257.Offset 140h: MBCSR - MemBIST Control Register (Sheet 1 of 3)

Description: MBCSR: Top level control register for DDR MemBIST.					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 140h **Offset End:** 143h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	START	**Start operation:** 1 => Set this bit to begin MemBIST execution. 0 => Hardware will clear this bit when MemBIST execution is completed.	N	0b	RWS

Table 16-257.Offset 140h: MBCSR - MemBIST Control Register (Sheet 2 of 3)

Description: MBCSR: Top level control register for DDR MemBIST.					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 140h **Offset End:** 143h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
30	PF	**Fail/Pass indicator:** Write to 0 when start MemBIST. Hardware will set to 1 when a failure is detected. 0 => Pass 1 => Fail	N	0b	RW
29	HALT	**Halt on Error:** 0 => Operation will not halt due to a detected error. 1 => Operation will halt after read-compare data error is detected. MemBIST will complete the current transaction before halting. This may result in multiple errors being logged.	N	0b	RW
28	ABORT	**MemBIST test abort**. When test abort bit is set, MBCSR bit 31 (Start operation, RWS) needs to be set to "0" at the same time to avoid restarting MemBIST. 0 => Normal operation. 1 => Need to abort the test during MemBIST operation. If there is any following Membist test after the abort test, bit [28] needs to be cleared. The Write to set MBCSR.abort must occur at least tRFC after the Write to set MBCSR.start. Otherwise subsequent MemBIST operations may fail.	N	0b	RW
27	SPARE	Reserved	N	00b	RO
26 :24	ALGO	000b: only support setting	N	000b	RW
23 :22	Reserved	Reserved	N	00b	RO
21 :20	CS	**Chip Select**[1:0] selection in MemBIST mode 01: select Rank 0 10: select Rank 1 00: Reserved 11: Reserved	N	00b	RW
19	INV	0b: only supported setting	N	0h	RW
18 :16	FX	**FIXED**: Fixed data pattern selection for MemBIST operation 000 => 0 001 => F 010 => A 011 => 5 100 => C 101 => 3 110 => 9 111 => 6	N	000b	RW
15	EN288	0b: only supported setting	N	0b	RW

Table 16-257.Offset 140h: MBCSR - MemBIST Control Register (Sheet 3 of 3)

Description: MBCSR: Top level control register for DDR MemBIST.					
View: PCI	BAR: SMRBASE	Bus:Device:Function: 0:0:0		Offset Start: 140h Offset End: 143h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14	MBDATA	**MBDATA**: Selects use of MBDATA for error log field for LFSR, Circular Shift and user defined data modes. This field has no effect on fixed data patterns. 0 => use MBDATA0/1/2/3/8 for failure data bit location accumulator. 1 => use MBDATA0/1/2/3/8 to log 5 failure addresses.	N	0b	RW
13	ABAR	0: only supported setting	N	0b	RW
12	ADIR	**ADIR:** Address decode direction 0 => Address increments 1 => Address decrements	N	0b	RW
11 :10	FAST	FAST Address sequencing 00: only supported setting	N	00b	RW
9 :08	DTYPE	Data type selection: 00 => Fixed data pattern, selected by MBCSR bits 18:16 01 => 144 bits user defined data 10 => Circular shift data based on Seed in MBLFSRSED 11 => LFSR data, seeded from 32 bit LFSR seed register. Note: Circular shift data and LFSR data type should not be used for single address operation (ATYPE = 01). Note: Circular shift data and LFSR data type only for 72-bit mode	N	00b	RW
7 :06	ATYPE	**Address type**: 00 => Reserved 01 => Single physical address operation, contained in MBADDR row/column/bank. 10 => start/end physical address range defined in MB_START_ADDR & MB_END_ADDR registers. 11 => full address range of the DIMM as defined in DRA/DRB registers which specifies the number of banks, rows, and columns. •	N	00b	RW
5 :04	CMD	**Command execution**: 00 => Read only without data comparison 01 => Write only 10 => Read with data comparison 11 => Write followed by Read with data comparison	N	00b	RW
3 :00	Reserved	Reserved	N	0000b	RO

16.5.1.36 Offset 144h: MBADDR - Memory Test Address Register

The register is used by MemBIST only when testing to a single memory location.

(MBCSR.atype = 2b'01)

Table 16-258.Offset 144h: MBADDR - Memory Test Address Register

Description:	MBADDR: Memory Test Address				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 144h **Offset End:** 147h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	ROW	Row Address 15:0	Y	0000h	RW
15	SPARE	Reserved. Must write as '0'	Y	0b	RW
14 :03	COL	Column Address BL8[14:3] <==> DRAM Column Address 15:11,9:3 BL4[14:3] <==> DRAM Column Address 14:11,9:2	Y	0000h	RW
2 :00	BA	Bank Address 2:0	Y	000b	RW

16.5.1.37 Offset 148h: MBDATA[0:9] - Memory Test Data Register

Table 16-259.Offset 148h: MBDATA[0:9] - Memory Test Data Register

Description:	MBADDR[0:9]: Memory Test Data				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 148h at 4h **Offset End:** 14Ch at 4h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	MBDATA	Usage varies by mode, refer to table below for details	Y	00000000h	RW

Reg	**Bit**	**Offset**	**Description by mode**			
			Fixed Data Pattern	**144 bit User Defined Pattern**	**Circular Shift**	**LFSR**
MBDATA9	31:0	**16Ch**	5th Fail address	User defined Late data [71:64] & Early data [71:64]	Word4 Circular shift data	LFSR random Late data [71:64] & Early data [71:64]
MBDATA8	31:0	**168h**	Late data [71:64] & Early data [71:64] Failure bit location accumulator	5th Fail address Or Late data [71:64] & Early data [71:64] Failure bit location accumulator	5th Fail address Or Late data [71:64] & Early data [71:64] Failure bit location accumulator	5th Fail address Or Late data [71:64] & Early data [71:64] Failure bit location accumulator

Reg	Bit	Offset	Description by mode			
			Fixed Data Pattern	144 bit User Defined Pattern	Circular Shift	LFSR
MBDATA7	31:0	164h	Fail address 4	User defined Late data [63:32]	DW3 Circular shift data	LFSR random Late data [63:32]
MBDATA6	31:0	160h	Fail address 3	User defined Late data [31:0]	DW2 Circular shift data	LFSR random Late data [31:0]
MBDATA5	31:0	15Ch	Fail address 2	User defined Early data [63:32]	DW1 Circular shift data	LFSR random Early data [63:32]
MBDATA4	31:0	158h	Fail address 1	User defined Early data [31:0]	DW0 Circular shift data	LFSR random Early data [31:0]
MBDATA3	31:0	154h	Late data [63:32] Failure bit location accumulator	Fail address 4 Or Late data [63:32] Failure bit location accumulator	Fail address 4 Or Late data [63:32] Failure bit location accumulator	Fail address 4 Or Late data [63:32] Failure bit location accumulator
MBDATA2	31:0	150h	Late data [31:0] Failure bit location accumulator	Fail address 3 Or Late data [31:0] Failure bit location accumulator	Fail address 3 Or Late data [31:0] Failure bit location accumulator	Fail address 3 Or Late data [31:0] Failure bit location accumulator
MBDATA1	31:0	14Ch	Early data [63:32] Failure bit location accumulator	Fail address 2 Or Early data [63:32] Failure bit location accumulator	Fail address 2 Or Early data [63:32] Failure bit location accumulator	Fail address 2 Or Early data [63:32] Failure bit location accumulator
MBDATA0	31:0	148h	Early data [31:0] Failure bit location accumulator	Fail address 1 Or Early data [31:0] Failure bit location accumulator	Fail address 1 Or Early data [31:0] Failure bit location accumulator	Fail address 1 Or Early data [31:0] Failure bit location accumulator

NOTE: In the later half part of data burst length 8 test, 144 bits user-defined data pattern will be repeat as the same sequence of burst length 4.

16.5.1.37.1 MBDATA Failure Address Mapping

To compress the failure address into 32 bits, bits that are always zero are removed from the logging. These removed bits include AutoPrecharge Column address [10] and least significant bits assumed by burst length.

Table 16-260.MBDATA Failure Address Register Correspondence to DRAM Address

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	see description below												
Bank			Row																Column and Chunk												

BL4: 1 bit chunk indicates the location of 2 failure burst data chunks.

The above Column plus Chunk is equal to DRAM column address as the following:

Table 16-261.BL4 Column and Chunk Correspondence to DRAM Address

Register Bit Location	12	11	10	9	8	7	6	5	4	3	2	1	0	
DRAM Col Address	14	13	12	11	9	8	7	6	5	4	3	2		
Data Chunk													1	X

- Where the auto-precharge address bit 10 is assumed zero.
- Since data is logged in 144 bits (two chunks), address bit zero is not needed.

BL8: 2 bit chunk indicates the location of 4 failure burst data chunks.

The above Column plus Chunk is equal to DRAM column address as the following:

Table 16-262.BL8 Column and Chunk Correspondence to DRAM Address

Register Bit Location	12	11	10	9	8	7	6	5	4	3	2	1	0	
DRAM Col Address	14	13	12	11	9	8	7	6	5	4	3			
Data Chunk												2	1	X

- Where the auto-precharge address bit 10 is assumed zero.
- Since data is logged in 144 bits (two chunks), Data chunk address bit zero is not needed.

16.5.1.38 Offset 19Ch: MB_START_ADDR - Memory Test Start Address Register

MB_END_ADDR row and column address must be larger than MB_START_ADDR row and column address in either increasing or deceasing address mode.

During FastX, FastY and FastXY operation, only one memory bank is tested. Specify the desired bank in MB_START_ADDR[2:0]. MB_END_ADDR[2:0] is ignored.

In XZY address mode, bank address wraps around either in the 4 banks or 8 banks case. In XZY mode, MB_END_ADDR bank address may be smaller than MB_START_ADDR bank address.

This register is only used when MBCSR.atype = 2b'10, and when MBCSR.algo is non-zero.

Table 16-263.Offset 19Ch: MB_START_ADDR - Memory Test Start Address Register

Description:	MB_START_ADDR: Memory Test Start Address				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 19Ch **Offset End:** 19Fh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	ROW	MemBIST Start Row Address 15:0	Y	0000h	RW
15	RESERVED	Reserved		0b	RO
14 :03	COL	MemBIST Start Column Address BL8[14:3] <==> DRAM Column Address 15:11,9:3 BL4[14:3] <==> DRAM Column Address 14:11,9:2	Y	0000h	RW
02 :00	BA	MemBIST Start Bank Address 2:0	Y	000b	RW

16.5.1.39 Offset 1A0h: MB_END_ADDR - Memory Test End Address Register

This register is only used when MBCSR.atype = 2b'10, and when MBCSR.algo is non-zero.

Table 16-264.Offset 1A0h: MB_END_ADDR - Memory Test End Address Register

Description:	MB_END_ADDR: Memory Test End Address				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1A0h **Offset End:** 1A3h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	ROW	MemBIST End Row Address 15:0	Y	0000h	RW
15	RESERVED	Reserved	N	0b	RO
14 :03	COL	MemBIST End Column Address BL8[14:3] <==> DRAM Column Address 15:11,9:3 BL4[14:3] <==> DRAM Column Address 14:11,9:2	Y	0000h	RW
02 :0	BA	MemBIST End Bank Address 2:0	Y	000b	RW

16.5.1.40 Offset 1A4h: MBLFSRSED - Memory Test Circular Shift and LFSR Seed Register

Note: If LFSR operation is selected, seed value of all 1s will NOT be able to generate random numbers. Pattern will remain all 1s.

Table 16-265.Offset 1A4h: MBLFSRSED - Memory Test Circular Shift and LFSR Seed Register

Description:	MBLFSRSED: Memory Test Circular Shift and LFSR Seed				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1A4h **Offset End:** 1A7h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	MBLFSRSED	MemBIST LFSR Seed This 32 bit register will be used as the initial data seed for LFSR or Circular shift data pattern.	Y	0h	RW

16.5.1.41 Offset 1A8h: MBFADDRPTR - Memory Test Failure Address Pointer Register

Table 16-266.Offset 1A8h: MBFADDRPTR - Memory Test Failure Address Pointer Register

Description:	MBFADDRPTR: Memory Test Failure Address Pointer Register				
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1A8h **Offset End:** 1ABh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	MBFADDRPTR	This 32 bit register designates which MemBIST failures to log in the available failure address locations. The default value of this register is zero. It means MemBIST always logs beginning with the first failure. If it is programmed to hex A (10 in decimal), MemBIST will log failures starting from the11th failure. The corresponding MB_ERR_DATA0/1/2/3 registers will log corrupted data in the first through fourth designated failure addresses. Note: this register does not affect the MBDATA failure bit location accumulators.	Y	00000000h	RW

16.5.1.42 Offset 1B0h: MB_ERR_DATA00 - Memory Test Error Data 0

Stores the first 32 bits of the 1st 144 bit failure data.

Table 16-267.Offset 1B0h: MB_ERR_DATA00 - Memory Test Error Data 0

Description:	MB_ERR_DATA00					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1B0h Offset End: 1B3h	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [31:0]		Y	00000000h	RW

16.5.1.43 Offset 1B4h: MB_ERR_DATA01 - Memory Test Error Data 0

Stores the second 32 bits of the 1st 144 bit failure data

Table 16-268.Offset 1B4h: MB_ERR_DATA01 - Memory Test Error Data 0

Description:	MB_ERR_DATA01					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1B4h Offset End: 1B7h	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [63:32]		Y	00000000h	RW

16.5.1.44 Offset 1B8h: MB_ERR_DATA02 - Memory Test Error Data 0

Stores the third 32 bits of the 1st 144 bit failure data

Table 16-269.Offset 1B8h: MB_ERR_DATA02 - Memory Test Error Data 0

Description:	MB_ERR_DATA02					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1B8h Offset End: 1BBh	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [31:0]		Y	00000000h	RW

16.5.1.45 Offset 1BCh: MB_ERR_DATA03 - Memory Test Error Data 0

Stores the fourth 32 bits of the 1st 144 bit failure data

Table 16-270.Offset 1BCh: MB_ERR_DATA03 - Memory Test Error Data 0

Description:	MB_ERR_DATA03					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0		**Offset Start:** 1BCh **Offset End:** 1BFh	
Size: 32 bit	**Default:** 00h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [63:32]		Y	00000000h	RW

16.5.1.46 Offset 1C0h: MB_ERR_DATA04 - Memory Test Error Data 0

Stores the last 16 bits of the 1st 144 bit failure data.

Table 16-271.Offset 1C0h: MB_ERR_DATA04 - Memory Test Error Data 0

Description:	MB_ERR_DATA04					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0		**Offset Start:** 1C0h **Offset End:** 1C1h	
Size: 16 bit	**Default:** 00h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DATA	Late failure data [71:64] & Early failure data [71:64]		Y	0000h	RW

16.5.1.47 Offset 1C4h: MB_ERR_DATA10 - Memory Test Error Data 1

Stores the first 32 bits of the 2nd 144 bit failure data

Table 16-272.Offset 1C4h: MB_ERR_DATA10 - Memory Test Error Data 1

Description:	MB_ERR_DATA10					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0		**Offset Start:** 1C4h **Offset End:** 1C7h	
Size: 32 bit	**Default:** 00h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [31:0]		Y	00000000h	RW

16.5.1.48 Offset 1C8h: MB_ERR_DATA11 - Memory Test Error Data 1

Stores the second 32 bits of the 2nd 144 bit failure data

Table 16-273.Offset 1C8h: MB_ERR_DATA11 - Memory Test Error Data 1

Description: MB_ERR_DATA11					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1C8h **Offset End:** 1CBh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [63:32]	Y	00000000h	RW

16.5.1.49 Offset 1CCh: MB_ERR_DATA12 - Memory Test Error Data 1

Stores the third 32 bits of the 2nd 144 bit failure data

Table 16-274.Offset 1CCh: MB_ERR_DATA12 - Memory Test Error Data 1

Description: MB_ERR_DATA12					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1CCh **Offset End:** 1CFh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [31:0]	Y	00000000h	RW

16.5.1.50 Offset 1D0h: MB_ERR_DATA13 - Memory Test Error Data 1

Stores the fourth 32 bits of the 2nd 144 bit failure data

Table 16-275.Offset 1D0h: MB_ERR_DATA13 - Memory Test Error Data 1

Description: MB_ERR_DATA13					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 1D0h **Offset End:** 1D3h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [63:32]	Y	00000000h	RW

16.5.1.51 Offset 1D4h: MB_ERR_DATA14 - Memory Test Error Data 1

Stores the last 16 bits of the 2nd 144 bit failure data

Table 16-276.Offset 1D4h: MB_ERR_DATA14 - Memory Test Error Data 1

Description: MB_ERR_DATA14					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1D4h **Offset End:** 1D5h	
Size: 16 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DATA	Late failure data [71:64] & Early failure data [71:64]	Y	0000h	RW

16.5.1.52 Offset 1D8h: MB_ERR_DATA20 - Memory Test Error Data 2

Stores the first 32 bits of the 3rd 144 bit failure data

Table 16-277.Offset 1D8h: MB_ERR_DATA20 - Memory Test Error Data 2

Description: MB_ERR_DATA20					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1D8h **Offset End:** 1DBh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [31:0]	Y	00000000h	RW

16.5.1.53 Offset 1DCh: MB_ERR_DATA21 - Memory Test Error Data 2

Stores the second 32 bits of the 3rd 144 bit failure data

Table 16-278.Offset 1DCh: MB_ERR_DATA21 - Memory Test Error Data 2

Description: MB_ERR_DATA21					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1DCh **Offset End:** 1DFh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [63:32]	Y	00000000h	RW

16.5.1.54 Offset 1E0h: MB_ERR_DATA22 - Memory Test Error Data 2

Stores the third 32 bits of the 3rd 144 bit failure data

Table 16-279.Offset 1E0h: MB_ERR_DATA22 - Memory Test Error Data 2

Description:	MB_ERR_DATA22					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1E0h Offset End: 1E3h	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [31:0]		Y	00000000h	RW

16.5.1.55 Offset 1E4h: MB_ERR_DATA23 - Memory Test Error Data 2

Stores the fourth 32 bits of the 3rd 144 bit failure data

Table 16-280.Offset 1E4h: MB_ERR_DATA23 - Memory Test Error Data 2

Description:	MB_ERR_DATA23					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1E4h Offset End: 1E7h	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [63:32]		Y	00000000h	RW

16.5.1.56 Offset 1E8h: MB_ERR_DATA24 - Memory Test Error Data 2

Stores the last 16 bits of the 3rd 144 bit failure data

Table 16-281.Offset 1E8h: MB_ERR_DATA24 - Memory Test Error Data 2

Description:	MB_ERR_DATA24					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1E8h Offset End: 1E9h	
Size: 16 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DATA	Late failure data [71:64] & Early failure data [71:64]		Y	0000h	RW

16.5.1.57 Offset 1ECh: MB_ERR_DATA30 - Memory Test Error Data 3

Stores the first 32 bits of the 4th 144 bit failure data

Table 16-282.Offset 1ECh: MB_ERR_DATA30 - Memory Test Error Data 3

Description:	MB_ERR_DATA30				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1ECh **Offset End:** 1EFh	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [31:0]	Y	00000000h	RW

16.5.1.58 Offset 1F0h: MB_ERR_DATA31 - Memory Test Error Data 3

Stores the second 32 bits of the 4th 144 bit failure data

Table 16-283.Offset 1F0h: MB_ERR_DATA31 - Memory Test Error Data 3

Description:	MB_ERR_DATA31				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1F0h **Offset End:** 1F4h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Early failure data [63:32]	Y	00000000h	RW

16.5.1.59 Offset 1F4h: MB_ERR_DATA32 - Memory Test Error Data 3

Stores the third 32 bits of the 4th 144 bit failure data

Table 16-284.Offset 1F4h: MB_ERR_DATA32 - Memory Test Error Data 3

Description:	MB_ERR_DATA32				
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1F4h **Offset End:** 1F7h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [31:0]	Y	00000000h	RW

16.5.1.60 Offset 1F8h: MB_ERR_DATA33 - Memory Test Error Data 3

Stores the fourth 32 bits of the 4th 144 bit failure data

Table 16-285.Offset 1F8h: MB_ERR_DATA33 - Memory Test Error Data 3

Description:	MB_ERR_DATA33					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1F8h Offset End: 1FBh	
Size: 32 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DATA	Late failure data [63:32]		Y	00000000h	RW

16.5.1.61 Offset 1FCh: MB_ERR_DATA34 - Memory Test Error Data 3

Stores the last 16 bits of the 4th 144 bit failure data.

Table 16-286.Offset 1FCh: MB_ERR_DATA34 - Memory Test Error Data 3

Description:	MB_ERR_DATA34					
View: PCI	BAR: SMRBASE		Bus:Device:Function: 0:0:0		Offset Start: 1FCh Offset End: 1FDh	
Size: 16 bit	Default: 00h				Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DATA	Late failure data [71:64] & Early failure data [71:64]		Y	0000h	RW

16.5.1.62 Offset 260h: DDRIOMC0 - DDR IO Mode Control Register 0

This register controls functionality of the DDRIO.

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, “Offset 14h: SMRBASE - System Memory RCOMP Base Address Register” on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

One function of DDRIOMC0 is to control the Voltage Crossing (VOX) analog control loop used to minimize any mismatch between the clock to output (Tco) on a low-to-high or high-to-low transition.

Table 16-287.Offset 260h: DDRIOMC0 - DDRIO Mode Register Control Register

Description:					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 260h **Offset End:** 263h	
Size: 32 bit	**Default:** 00000078h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 13	Reserved	Reserved	N	0000000000000000000b	RO
12 9	DQVOXADJ	Bits to configure DQ buffer tco balancing	Y	0000b	RW
8 8	DDRVOXCTL1	Combine this bit with DDRVOXCTL0 (defined below) Encodings: 00 : DQ and CA buffers are in VOX Cross Reference Mode 01: Bypass DQ and CA VOX Cross Reference Mode (default) 10: VOX Bypass Mode 11: Reset VOX Mode	Y	0b	RW
7 7	Reserved	Reserved	N	0b	RO
6 4	Reserved	Reserved	N	111b	RW
3 3	DDRVOXCTL0	This is the least significant bit of DDRVOXCTL. For encoding details, see DDRVOXCTL1 above	Y	1b	RW
2 0	Reserved	Reserved	Y	000b	RW

16.5.1.63 Offset 264h: DDRIOMC1 - DDR IO Mode Control Register 1

This register controls functionality of the DDRIO.

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, “Offset 14h: SMRBASE - System Memory RCOMP Base Address Register” on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

Table 16-288.Offset 264h: DDRIOMC1 - DDRIO Mode Register Control Register 1 (Sheet 1 of 2)

<table>
<tr><td colspan="6">Description: DDRIOMC1: DDRIO Mode Control Register 1</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: SMRBASE</td><td>Bus:Device:Function: 0:0:0</td><td colspan="2">Offset Start: 264h
Offset End: 267h</td></tr>
<tr><td>Size: 32 bit</td><td colspan="2">Default: 52520000h</td><td></td><td colspan="2">Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>31 24</td><td>CASLEW</td><td>CASLEW: The digital slew override 8-bit control allow for balancing of pull-up and pull-down slew rates T for CA/CLK buffers. The format of these controls and recommended reset value is given below:
<table><tr><td>Bits</td><td>Function</td></tr><tr><td>7</td><td>DDR2Selection.
DDR2 = 0</td></tr><tr><td>6:5</td><td>Fast Corner falling slew rate trim</td></tr><tr><td>4:2</td><td>Slow Corner falling slew rate trim</td></tr><tr><td>1:0</td><td>Fast corner rising slew rate trim</td></tr></table></td><td>Y</td><td>01010010b</td><td>RW</td></tr>
<tr><td>23 16</td><td>DQSLEW</td><td>DQSLEW: The digital slew override 8-bit control allow for balancing of pull-up and pull-down slew rates T for CA/CLK buffers. The format of these controls and recommended reset value is given below:
<table><tr><td>Bits</td><td>Function</td></tr><tr><td>7</td><td>DDR2 Selection.
DDR2 = 0</td></tr><tr><td>6:5</td><td>Fast Corner falling slew rate trim</td></tr><tr><td>4:2</td><td>Slow Corner falling slew rate trim</td></tr><tr><td>1:0</td><td>Fast corner rising slew rate trim</td></tr></table></td><td>Y</td><td>01010010b</td><td>RW</td></tr>
<tr><td>15 7</td><td>Reserved</td><td>Reserved</td><td>N</td><td>000000000b</td><td>RO</td></tr>
</table>

Table 16-288.Offset 264h: DDRIOMC1 - DDRIO Mode Register Control Register 1 (Sheet 2 of 2)

<table>
<tr><td colspan="6">Description: DDRIOMC1: DDRIO Mode Control Register 1</td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: SMRBASE</td><td colspan="2">Bus:Device:Function: 0:0:0</td><td>Offset Start: 264h
Offset End: 267h</td></tr>
<tr><td>Size: 32 bit</td><td colspan="2">Default: 52520000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>6 5</td><td>DEMPDQ</td><td>De-emphasis mode select bit for DQ/DQS pins. This mode can be used to reduce power and enhance data eyes. When de-emphasis is enable for a given group of I/Os, subsequent driver values that are the same have their strength reduced by half
It is recommended that this be controllable by the BIOS in case there are unwanted side effects of this feature.
<table><tr><td>Encoding</td><td>Description</td></tr><tr><td>00</td><td>Disabled</td></tr><tr><td>01</td><td>Weakly Enabled</td></tr><tr><td>10</td><td>Full Enabled</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>4 3</td><td>DEMPCA</td><td>De-emphasis mode select bit for command/clock pins. This mode can be used to reduce power and enhance data eyes. When de-emphasis is enable for a given group of I/Os, subsequent driver values that are the same have their strength reduced by half.
It is recommended that this be controllable by the BIOS in case there are unwanted side effects of this feature. For instance, de-emphasis should be off before entering self-refresh mode of the DRAM to prevent the CKE from exceeding the JEDEC threshold once self-refresh is entered.
<table><tr><td>Encoding</td><td>Description</td></tr><tr><td>00</td><td>Disabled</td></tr><tr><td>01</td><td>Weakly Enabled</td></tr><tr><td>10</td><td>Full Enabled</td></tr><tr><td>Others</td><td>Reserved</td></tr></table></td><td>Y</td><td>00b</td><td>RW</td></tr>
<tr><td>2 2</td><td>Reserved</td><td>Reserved</td><td>Y</td><td>0b</td><td>RW</td></tr>
<tr><td>1 0</td><td>FASTSLEW</td><td>bit[0] controls the control bits
bit[1] controls the data bits</td><td>Y</td><td>00b</td><td>RW</td></tr>
</table>

16.5.1.64 Offset 268h: DDRIOMC2 - DDR IO Mode Control Register 2

This register controls functionality of the DDRIO.

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

Acceptable values for legoverride[4:0] are 0-30d

Example(R_DQ, dynamic): Assume Rext = 249 ohms

R_DQ = Rext*2/(31-legoverride[4:0])

When legoverride[4:0] = 00000, R_DQ = (249*2)/(31-0) = 16.1 ohms

When legoverride[4:0] = 11110, R_DQ = (249*2)/(31-30) = 498 ohms

Table 16-289.Legoverride details

Mode (select via legoverride[5:4])	Legoverride[9:6], [4:0]	Details
static: legoverride[5:4] = 10b	[9:6] static impedance control setting of CMD/ADD/CLK	See gray code below (valid for 1 to 10)
	[3:0] static impedance control setting of DQ/DM/DQS	See gray code below (valid for 1 to 10)
dynamic: legoverride[5] = 0b	[9:6] dynamic impedance control setting of CMD/ADD/CLK	R_CA = R_DQ * 6.154 / Legoveride[9:6] or R_min_CA (Ohm) Where, Rext is the external R connected to DDR_DRVCRES pin and Rmin is impedance when all the driver legs are selected. Rmin is process/voltage/temp dependent.
	[4:0] dynamic impedance control setting of DQ/DM/DQS	R_DQ = Rext*2 / (31- Legoverride[3:0]) or R_min_DQ (Ohm) Where, Rext is the external R connected to DDR_DRVCRES pin and Rmin is impedance when all the driver legs are selected. Rmin is process/voltage/temp dependent.

Table 16-290.Legoverride - Gray code

CSR value for static impedance control (binary)	Desired value for impedance control (Decimal)
1000	15
1001	14
1011	13
1010	12
1110	11
1111	10
1101	9

Table 16-290.Legoverride - Gray code

CSR value for static impedance control (binary)	Desired value for impedance control (Decimal)
1100	8
0100	7
0101	6
0111	5
0110	4
0010	3
00011	2
0001	1
0000	0

:

Table 16-291.Offset 268h: DDRIOMC2 - DDRIO Mode Control Register 2

Description: DDRIOMC2: DDRIO Mode Control Register 2					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 268h **Offset End:** 26Bh	
Size: 32 bit	**Default:** 039E6000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	Reserved	Reserved	N	0000b	RO
27 26	PHSEL	Core phase to Command/Address relationship.	Y	00b	RW
25 16	LEGOVERRIDE	Digital Impedance Control for RCOMP of DDR pads. See Legoverride table above. Do not use the Default setting Please refer to Section 11.4.6, “RCOMP” for more details.	Y	1110011110b	RW
15 15	FIFOWPTRCLR	This bit clears the DDRIO Receive FIFO read and write pointers. The write pointer of this FIFO is generated by the DDRIO logic based on DQS while the read pointer is generated by the memory controller. The DDRIO receive FIPO read/write pointers need to be cleared after DCAL or Mbist operations are completed and before issuing any functional DRAM R/W operations. Unlike SDRC.DDRRFRS this register will reset only the read/write pointers of the DDRIO receive FIFO. It will not reset the DLL’s. Please see Section 16.1.1.45, “Offset 88h: SDRC – DDR SDRAM Secondary Control Register” for more details.	N	0b	RW
14 :12	MASTCNTL	Coarse delay of DQS Master DLL	Y	110b	RW
11 :00	Reserved	Reserved	N	000000000000b	RO

16.5.1.65 Offset 284h: WL_CNTL[4:0] - Write Levelization[4:0] Control Register

This register controls functionality of Write Levelization (WL). The EP80579 implements 5 CSR's to control WL. The mapping of each of these CSR's to the byte lane is shown in Table 16-292.

Table 16-292.Mapping of DQ and DQS/# byte lanes to WL_CNTL[4:0] CSR's

CSR Name	WL_CNTL CSR #	Generate DQS/# Pulse [CSR bits]	Latch DQ Feedback Data [CSR bits]
Byte 0 DQS/#[0] DQ[7:0]	WL_CNTL[0]	ODD_DRV_WL_PULSE	ODD_WL_DATA[7:0]
Byte 1 DQS/#[1] DQ[15:8]		EVEN_DR_WL_PULSE	EVEN_WL_DATA[7:0]
Byte 2 DQS/#[2] DQ[23:16]	WL_CNTL[1]	ODD_DRV_WL_PULSE	ODD_WL_DATA[7:0]
Byte 3 DQS/#[0] DQ[31:24]		EVEN_DR_WL_PULSE	EVEN_WL_DATA[7:0]
Byte 4 DQS/#[0] DQ[39:32]	WL_CNTL[2]	ODD_DRV_WL_PULSE	ODD_WL_DATA[7:0]
Byte 5 DQS/#[0] DQ[47:40]		EVEN_DR_WL_PULSE	EVEN_WL_DATA[7:0]
Byte 6 DQS/#[6] DQ[55:48]	WL_CNTL[3]	ODD_DRV_WL_PULSE	ODD_WL_DATA[7:0]
Byte 7 DQS/#[7] DQ[63:56]		EVEN_DR_WL_PULSE	EVEN_WL_DATA[7:0]
Byte 8 DQS/#[8] DQ[71:64]	WL_CNTL[4]	ODD_DRV_WL_PULSE	ODD_WL_DATA[7:0]

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, "Offset 14h: SMRBASE - System Memory RCOMP Base Address Register" on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

Table 16-293.Offset 284h: WL_CNTL[4:0] - Write Levelization Control Register

Description: WL_CNTL[4:0]: Write Levelization Control Register					
View: PCI	**BAR:** SMRBASE	**Bus:Device:Function:** 0:0:0		**Offset Start:** 284h at 4h **Offset End:** 294h at 4h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved	N	0000h	RO
15 :14	Reserved	Reserved	N	00b	RW
13 :12	Reserved	Reserved	N	00b	RO
11 :8	WL_CNTRL	Delay Select See Table 16-294	Y	0000b	RW
7 :2	WDLL_CNTL	Length controls for Slave Write DLL (WDLL). A delay of 0 up to 3/8 of clk1x can be programmed using this CSR.	Y	00000b	RW
1 :1	WDLL_CLKG	Control bit for Clock gating of DQ/DQS. 0– Disable clock gating for DQ/DQS 1– Enable clock gating for DQ/DQS Note: for WL_CNTL[4], WDLL_CLKG must be equal to 0	Y	0b	RW
0 :0	BYP_WDLL	Bypass Write DLL. This bit is used only for centering DQS to the DQ eye. For write leveling, see Table 16-294. 0 – Bypass DLL 1 – Output with WDLL Before enabling/setting this bit to 1, software needs to first program the appropriate values in DRAMDLLC.SLVLEN & WL_CNTL[x].WDLL_CNTL.	Y	0b	RW

16.5.1.65.1 Control of delay for DQ/DQs

There are two methods to control delay:

- Adding portions of a clock cycle
- Adding portions of a clock cycle as well as delay from the WDLL

The full control is encoded by combining the appropriate bit of WDLL_MISC.WL_PHSEL_MODE with WL_CNTL[x].WL_CNTRL as shown in Table 16-294

Table 16-294. Delay of DQ/DQS

WDLL_MISC.WL_PHSEL_MODE, WL_CNTL[x].WL_CNTRL	Delay
00xxx	0
01000	WDLL_DLY
01011	1/4 CLK + WDLL_DLY
01100	1/2 CLK + WDLL_DLY
01101	3/4 CLK + WDLL_DLY
10xxx	0
11001	1/4 CLK
11000	1/2 CLK
11011	3/4 CLK
11100	1 CLK

16.5.1.65.2 Formula to calculate delay through DLL

delay_uncomp = 100 ps (approximate value)
delay_element = (1/4 CLK period - delay_uncomp) / (DDRIOMC2.MASTCNTL + 1/2)
WDLL_DLY = (delay_element * WL_CNTL.WDLL_CNTL)/8

16.5.1.66 Offset 298h: WDLL_MISC - DLL Miscellaneous Control

This register controls miscellaneous functions of the DDRIO DLL.

This CSR is in the memory-mapped IO region of Bus 0, Device 0, Function 0 of the memory controller. The SMRBASE register described in Section 16.1.1.9, “Offset 14h: SMRBASE - System Memory RCOMP Base Address Register” on page 395, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The value for BAR for all registers in this section is BAR14h.

:

Table 16-295.Offset 298h: WDLL_MISC - DLL Miscellaneous Control

Description: WDLL_MISC- DLL Miscellaneous Control					
View: PCI	**BAR:** SMRBASE		**Bus:Device:Function:** 0:0:0	**Offset Start:** 298h **Offset End:** 29Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :25	Reserved	Reserved	N	0000000b	RO
24 :24	WLCKDLY	0: delay ECC/DQS[8]/DQS_L[8] only, clocks not delayed 1: delay ECC/DQS[8]/DQS_L[8] and CK[2:0]/CK_L[2:0] (Normal setting for DDR2)	Y	0b	RW
23 :23	Reserved	Reserved	N	0b	RO
22 :16	WL_PHSEL_MODE	See Table 16-294 for DQ/DQS Connectivity: [22] CS, ODT, CKE [21] CK[5:3], CK_L[5:3] [20] WL_CNTL[0] (DQ[15:0], DQS/DQS_L[1:0]) [19] WL_CNTL[1] (DQ[31:16], DQS/DQS_L[3:2]) [18] WL_CNTL[4] (ECC[7:0], DQS/DQS_L[8],CK[2:0], CK_L[2:0]) [17] WL_CNTL[2] (DQ[47:32], DQS/DQS_L[5:4]) [16] WL_CNTL[3] (DQ[63:48], DQS/DQS_L[7:6])	Y	0000000b	RW
15 :12	Reserved	Reserved	N	0000000b	RO
11 :8	WL_CNTRL	Delay select for CK[5:3] and CK_L[5:3]: 0xxx: no delay 1001: delay 1/4 clk1x 1000: delay 1/2 clk1x 1011: delay 3/4 clk1x 1100: delay 1 clk1x OthersReserved	Y	0000b	RW
7 :4	WL_CNTRL_A	Delay select for CS, ODT and CKE 0xxx: no delay 1001: delay 1/4 clk1x 1000: delay 1/2 clk1x 1011: delay 3/4 clk1x 1100: delay 1 clk1x OthersReserved	Y	0000b	RW
3 :3	Reserved	Reserved	N	0b	RO
2 :0	WL_CMD_DLY	Reserved to Intel Encoded additional delay for CS, CKE, ODT Delay introduced = (~100ps * WL_CMD_DLY)	Y	000b	RW

16.6 Memory Mapped I/O for EDMA Registers

The EDMA is a reuse module capable of 64-bit addressing on both source and destination interfaces.

For the EP80579 the EDMA will only be used in 32-bit addressing mode

This section describes the memory-mapped registers for the EDMA Controller. The EDMALBAR register, described in Section 16.3.1.9, “Offset 10h: EDMALBAR - EDMA Low Base Address Register”, provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

The BAR value for all registers in this section is BAR10h.

Each DMA channel consists of twelve 32-bit registers contiguous in memory mapped address space. The first of the four sets is described in detail, the others are copies at different offsets for the other channels. The abbreviations for the other channel register names replace the 0 at the end of the name with the appropriate channel number 1, 2, or 3.

Table 16-296.Bus 0, Device 1, Function 0: Summary of EDMA Configuration Registers Mapped Through EDMALBAR Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: CCR0 - Channel 0 Channel Control Register" on page 653	00000000h
04h	07h	"Offset 04h: CSR0 - Channel 0 Channel Status Register" on page 656	00000000h
08h	0Bh	"Offset 08h: CDAR0 - Channel 0 Current Descriptor Address Register" on page 657	00000000h
0Ch	0Fh	"Offset 0Ch: CDUAR0 - Channel 0 Current Descriptor Upper Address Register" on page 658	00000000h
10h	13h	"Offset 10h: SAR0 - Channel 0 Source Address Register" on page 658	00000000h
14h	17h	"Offset 14h: SUAR0 - Channel 0 Source Upper Address Register" on page 659	00000000h
18h	1Bh	"Offset 18h: DAR0 - Channel 0 Destination Address Register" on page 659	00000000h
1Ch	1Fh	"Offset 1Ch: DUAR0 - Channel 0 Destination Upper Address Register" on page 660	00000000h
20h	23h	"Offset 20h: NDAR0 - Channel 0 Next Descriptor Address Register" on page 661	00000000h
24h	27h	"Offset 24h: NDUAR0 - Channel 0 Next Descriptor Upper Address Register" on page 662	00000000h
28h	2Bh	"Offset 28h: TCR0 - Channel 0 Transfer Count Register" on page 662	00000000h
2Ch	2Fh	"Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register" on page 663	00000000h
40h	43h	"Offset 40h: CCR1 - Channel 1 Channel Control Register" on page 665	00000000h
44h	47h	"Offset 44h: CSR1 - Channel 1 Channel Status Register" on page 665	00000000h
48h	4Bh	"Offset 48h: CDAR1 - Channel 1 Current Descriptor Address Register" on page 665	00000000h
4Ch	4Fh	"Offset 4Ch: CDUAR1 - Channel 1 Current Descriptor Upper Address Register" on page 666	00000000h
50h	53h	"Offset 50h: SAR1 - Channel 1 Source Address Register" on page 666	00000000h
54h	57h	"Offset 54h: SUAR1 - Channel 1 Source Upper Address Register" on page 666	00000000h
58h	5Bh	"Offset 58h: DAR1 - Channel 1 Destination Address Register" on page 667	00000000h
5Ch	5Fh	"Offset 5Ch: DUAR1 - Channel 1 Destination Upper Address Register" on page 667	00000000h
60h	63h	"Offset 60h: NDAR1 - Channel 1 Next Descriptor Address Register" on page 667	00000000h
64h	67h	"Offset 64h: NDUAR1 - Channel 1 Next Descriptor Upper Address Register" on page 668	00000000h
68h	6Bh	"Offset 68h: TCR1 - Channel 1 Transfer Count Register" on page 668	00000000h
6Ch	6Fh	"Offset 6Ch: DCR1 - Channel 1 Descriptor Control Register" on page 668	00000000h
80h	83h	"Offset 80h: CCR2 - Channel 2 Channel Control Register" on page 669	00000000h
84h	87h	"Offset 84h: CSR2 - Channel 2 Channel Status Register" on page 669	00000000h
88h	8Bh	"Offset 88h: CDAR2: Channel 2 Current Descriptor Address Register" on page 669	00000000h
8Ch	8Fh	"Offset 8Ch: CDUAR2 - Channel 2 Current Descriptor Upper Address Register" on page 670	00000000h
90h	93h	"Offset 90h: SAR2 - Channel 2 Source Address Register" on page 670	00000000h
94h	97h	"Offset 94h: SUAR2 - Channel 2 Source Upper Address Register" on page 670	00000000h
98h	9Bh	"Offset 98h: DAR2 - Channel 2 Destination Address Register" on page 671	00000000h
9Ch	9Fh	"Offset 9Ch: DUAR2 - Channel 2 Destination Upper Address Register" on page 671	00000000h
A0h	A3h	"Offset A0h: NDAR2 - Channel 2 Next Descriptor Address Register" on page 671	00000000h
A4h	A7h	"Offset A4h: NDUAR2 - Channel 2 Next Descriptor Upper Address Register" on page 672	00000000h

Table 16-296.Bus 0, Device 1, Function 0: Summary of EDMA Configuration Registers Mapped Through EDMALBAR Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
A8h	ABh	"Offset A8h: DCR2 - Channel 2Transfer Control Register" on page 672	00000000h
ACh	AFh	"Offset ACh: DCR2 - Channel 2 Descriptor Control Register" on page 672	00000000h
C0h	C3h	"Offset C0h: CCR3 - Channel 3 Channel Control Register" on page 673	00000000h
C4h	C7h	"Offset C4h: CSR3 - Channel 3 Channel Status Register" on page 673	00000000h
C8h	CBh	"Offset C8h: CDAR3 - Channel 3 Current Descriptor Address Register" on page 673	00000000h
CCh	CFh	"Offset CCh: CDUAR3 - Channel 3 Current Descriptor Upper Address Register" on page 674	00000000h
D0h	D3h	"Offset D0h: SAR3 - Channel 3 Source Address Register" on page 674	00000000h
D4h	D7h	"Offset D4h: SUAR3 - Channel 3 Source Upper Address Register" on page 674	00000000h
D8h	DBh	"Offset D8h: DAR3 - Channel 3 Destination Address Register" on page 675	00000000h
DCh	DFh	"Offset DCh: DUAR3 - Channel 3 Destination Upper Address Register" on page 675	00000000h
E0h	E3h	"Offset E0h: NDAR3 - Channel 3 Next Descriptor Address Register" on page 675	00000000h
E4h	E7h	"Offset E4h: NDUAR3 - Channel 3 Next Descriptor Upper Address Register" on page 676	00000000h
E8h	EBh	"Offset E8h: TCR3 - Channel 3 Transfer Count Register" on page 676	00000000h
ECh	EFh	"Offset ECh: DCR3 - Channel 3 Descriptor Control Register" on page 677	00000000h
100h	103h	"Offset 100h: DCGC - EDMA Controller Global Command" on page 677	00000000h
104h	107h	"Offset 104h: DCGS - EDMA Controller Global Status" on page 678	00000000h

16.6.1 Register Details

16.6.1.1 Offset 00h: CCR0 - Channel 0 Channel Control Register

The Channel Control Register (CCR) is cleared to zero on power-on or system reset. The CCR specifies the overall operating environment for the channel. Software initializes this register only after initializing the chain descriptors in system memory, and the Next Address registers as pointer to the first chain descriptor in memory. CCR can be written when the DMA channel is active. The CCR is a read/write register.

Table 16-297.Offset 00h: CCR0 - Channel 0 Channel Control Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	Reserved	Reserved		000000h	

Table 16-297.Offset 00h: CCR0 - Channel 0 Channel Control Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	CRSM	Channel Resume: 0 = Cleared when: • The channel completes a DMA transfer and the Next Descriptor Address Register is not zero. In this case, the channel proceeds to the next descriptor in the chain (resumes). • The channel is idle or the channel completes a DMA transfer and the Next Descriptor Address Register is zero. 1 = Causes the channel to resume chaining by re-reading the current descriptor located in local system memory and reloading the Next Descriptor Address Register when the channel is idle (the Channel Active bit in the CSR is clear) or when the channel completes execution of the current descriptor. Once set, software cannot clear this bit. The IMCH prevents this bit from being set when either the stopped or aborted bit is set in the CSR. Software must clear the CSR stopped and aborted bits before attempting to resume the current descriptor chain. If the CSR end of chain bit was set, the DMA channel clears the end of chain bit when the current descriptor chain resumes. Refer to Chapter 12.0, "Enhanced Direct Memory Access Controller (EDMA)" for details on the suspend and resume function.		0b	RWS
02	STPDMA	**Stop:** 0 = Cleared only by the IMCH, once the Channel Active bit is cleared and the DMA Stopped bit is set. 1 = Causes the current DMA transfer to stop. The channel does not request the bus on the source side. Any data in the queue is emptied to the destination side, and all relevant bits in the CCR (bits 03:00) and CSR (Channel Active bit) are cleared. This bit has priority over the Suspend DMA bit. Once set, this bit cannot be cleared by the software. Software must be very careful in setting this bit since any DMA transfer, once stopped, cannot be restarted from that point.		0b	RWS
01	SUSDMA	**Suspend:** This has no effect on the Channel Active bit. 0 = Software clears this bit once the DMA Suspended bit is set. Clearing this bit restarts the DMA transfer from the point it was suspended, and clears the DMA Suspended bit in the CSR. Refer to Chapter 12.0, "Enhanced Direct Memory Access Controller (EDMA)" of for details on the DMA suspend function. 1 = Allows the current descriptor to finish, but suspends channel chaining. The channel continues to request the bus on the source side for the current descriptor. When the data in the queue for this descriptor is emptied to the destination side, the channel sets the DMA Suspended bit in the CSR.		0b	RW

Table 16-297.Offset 00h: CCR0 - Channel 0 Channel Control Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
00	STRTDMA	**Start:** 0 = Cleared by the IMCH when the DMA transfer is complete, when the DMA is stopped by software, or when the DMA encounters any unrecoverable error. The IMCH prevents this bit from being set when the stopped or aborted bit is set in the CSR. The DMA channel must be idle and software must clear the CSR before starting the DMA channel with a new descriptor chain. 1 = Channel is enabled for DMA transfer. Once set, this bit cannot be cleared by software.		0b	RWS

16.6.1.2 Offset 04h: CSR0 - Channel 0 Channel Status Register

The Channel Status Register (CSR) contains status flags that indicate the channel status. The register is read by application software to get the current channel status and to examine the source of an interrupt. Table 16-298 shows the format for the CSR.

Table 16-298.Offset 04h: CSR0 - Channel 0 Channel Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 06	Reserved	Reserved		0000000h	
05	CACTV	**Channel Active:** 0 = Channel is inactive and available to be configured for DMA transfer by software. 1 = Set by the IMCH, indicates the channel is in use and actively performing DMA data transfers. The channel active flag is set by the IMCH when: • Software initiates a DMA transfer by setting the Start bit of CCR and the DMA channel in response loads the chain descriptor from the local system memory • Software initiates a DMA transfer by setting the Channel Resume bit of the CCR and the NDAR/ NDUAR point to a legal non-null address in memory.		0b	RO
04	DABRT	**Aborted:** 0 = Software clears this bit by writing a 1 to the bit location. 1 = Indicates that the current DMA transfer for this channel encountered an unrecoverable error. If the Aborted Interrupt Enable bit in the DCR is set, this generates an interrupt to the processor. Software polls this bit if an interrupt is not enabled. Error details are logged in the DMA_FERR and DMA_NERR registers.		0b	RWC
03	DSTP	**Stopped:** 0 = Software clears this bit by writing a 1 to the bit location. 1 = Indicates that the current transfer for this channel has been stopped by software setting the Stop bit in the CCR. If the Stopped Interrupt Enable bit in the DCR is set, this generates an interrupt to the processor. Software can use this bit for polling if interrupts are not enabled.		0b	RWC
02	DSUS	**Suspended:** 0 = Cleared when software clears the Suspend bit in the CCR. 1 = Indicates that the current transfer for this channel has been stopped by software setting the Suspend bit in the CCR. If the Suspended Interrupt Enable bit in the DCR is set, this generates an interrupt to the processor. Software can use this bit for polling if interrupts are not enabled.		0b	RO

Table 16-298.Offset 04h: CSR0 - Channel 0 Channel Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	EOT	**End of Transfer:** 0 = Software clears this bit by writing a '1' to the bit location. 1 = Indicates that the channel has successfully completed an error-free DMA transfer of at least one descriptor. If the End of Transfer Interrupt Enable bit in the DCR is set, this generates an interrupt to the processor. Software can use this bit for polling if interrupts are not enabled.		0b	RWC
00	EOC	**End of Chain:** 0 = Software clears this bit by writing a '1' to the bit location. 1 = Indicates that the channel has successfully completed an error-free DMA transfer, and it is the last descriptor in a chain descriptor. If the End of Chain Interrupt Enable bit in the DCR is set, this generates an interrupt to the processor. Software can use this bit for polling if interrupts are not enabled.		0b	RWC

16.6.1.3 Offset 08h: CDAR0 - Channel 0 Current Descriptor Address Register

The Current Descriptor Address Register (CDAR) contains the lower 32-bit address of the current chain descriptor in local system memory. This register is loaded by the IMCH when a new chain descriptor is read. All chain descriptors are aligned on an eight double-word (32-bit) boundary.

Table 16-299.Offset 08h: CDAR0 - Channel 0 Current Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 05	CDADD	**Current Descriptor Address:** Lower 32 bits of the local system memory address of the current chain descriptor that is read by the channel. The descriptor address must be eight double-word aligned.		0000000h	RO
04 : 00	Reserved	Reserved		00h	

16.6.1.4 Offset 0Ch: CDUAR0 - Channel 0 Current Descriptor Upper Address Register

The upper address will not be used in the EP80579, which is limited to 32bit addressing.

The Current Descriptor Upper Address Register (CDUAR) contains the upper 32-bit address of the current chain descriptor in local system memory. This register is loaded by the IMCH when a new chain descriptor is read.

Note: Because the EP80579 supports 32 bit addressing only, this register needs to be set to "0" at all times.

Table 16-300.Offset 0Ch: CDUAR0 - Channel 0 Current Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 0Ch **Offset End:** 0Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CDUAR0	**Current Descriptor Address:** The upper 32-bit local system memory address of the current chain descriptor that is read by the channel.		0000000h	RO

16.6.1.5 Offset 10h: SAR0 - Channel 0 Source Address Register

The Source Address Register (SAR) contains the lower 32-bit source address for the current DMA transfer. This register is loaded by the IMCH when the source address field of a new chain descriptor is read.

Table 16-301.Offset 10h: SAR0 - Channel 0 Source Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	SAR0	**Current Source Address:** The lower 32-bit source memory address for the current DMA transfer.		0000000h	RO

16.6.1.6 Offset 14h: SUAR0 - Channel 0 Source Upper Address Register

The upper address will not be used in the EP80579, which is limited to 32bit addressing

The Source Upper Address Register (SUAR) contains the upper 32-bit source address for the current DMA transfer. This register is loaded by the IMCH when the source upper address field of a new chain descriptor is read.

Note: Because the EP80579 supports 32 bit addressing only, this register needs to be set to "0" at all times.

Table 16-302.Offset 14h: SUAR0 - Channel 0 Source Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	SUAR0	**Current Source Address:** The upper 32-bit source memory address for the current DMA transfer.		0000000h	RO

16.6.1.7 Offset 18h: DAR0 - Channel 0 Destination Address Register

The Destination Address Register (DAR) contains the lower 32-bit destination address for the current DMA transfer. This register is loaded by the IMCH when the destination address field of a new chain descriptor is read.

Table 16-303.Offset 18h: DAR0 - Channel 0 Destination Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 18h **Offset End:** 1Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DAR0	**Current Destination Address:** The lower 32-bit destination memory address for the current DMA transfer.		0000000h	RO

16.6.1.8 Offset 1Ch: DUAR0 - Channel 0 Destination Upper Address Register

The upper address will not be used in the EP80579, which is limited to 32bit addressing.

The Destination Upper Address Register (DUAR) contains the upper 32-bit destination address for the current DMA transfer. This register is loaded by the IMCH when the destination upper address field of a new chain descriptor is read.

Note: Because the EP80579 supports 32 bit addressing only, this register needs to be set to “0” at all times.

Table 16-304.Offset 1Ch: DUAR0 - Channel 0 Destination Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 1Ch **Offset End:** 1Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DUAR0	**Current Destination Address:** The upper 32-bit destination memory address for the current DMA transfer.		0000000h	RO

16.6.1.9 Offset 20h: NDAR0 - Channel 0 Next Descriptor Address Register

The Next Descriptor Address Register (NDAR) contains the lower 32-bit address of the next descriptor chain in the local system memory. This register is loaded when the next descriptor address field of a new chain descriptor is read. Additionally, software writes this register with the address of the first chain descriptor in local memory. All chain descriptors are required to be aligned on an eight double-word (32-bit) boundary or the CMI flags an error.

Note: Software must make sure that the Start bit in the CCR and the Channel Active bit in the CSR are clear prior to writing to the NDAR. The IMCH prevents writing to this register when these bits are not clear. Writing zero into the NDAR and NDUAR by software does not start a DMA transfer.

Table 16-305.Offset 20h: NDAR0 - Channel 0 Next Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 20h **Offset End:** 23h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	NDLADD	**Next Descriptor Lower Address:** Lower 32 bits of the local system memory address of the next chain descriptor in memory to be read by the channel. The address must be aligned on an eight DWord (32-bit) boundary or else the IMCH flags an error. This field can only be written when the Start bit in the CCR and the Channel Active bit in the CSR are clear.		0000000h	RWL

16.6.1.10 Offset 24h: NDUAR0 - Channel 0 Next Descriptor Upper Address Register

The upper address will not be used in the EP80579, which is limited to 32bit addressing.

The Next Descriptor Upper Address Register (NDUAR) contains the upper 32-bit address of the next descriptor chain in the local system memory. This register is loaded when the next descriptor address field of a new chain descriptor is read. Additionally, software writes this register with the address of the first chain descriptor in local memory.

Note: Software must make sure that the Start bit in the CCR and the Channel Active bit in the CSR are clear prior to writing to the Next Descriptor Upper Address Register (NDAR). The IMCH prevents writing to this register when these bits are not clear. Writing zero into the NDAR and NDUAR by application software does not start a DMA transfer.

Note: Because the EP80579 supports 32 bit addressing only, this register needs to be set to “0” at all times.

Table 16-306.Offset 24h: NDUAR0 - Channel 0 Next Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 24h **Offset End:** 27h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	NDUADD	**Next Descriptor Upper Address:** The upper 32-bit address of the next descriptor chain in memory to be read by the channel. This field can only be written when the Start bit in the CCR and the Channel Active bit in the CSR are clear.		0000000h	RWL

16.6.1.11 Offset 28h: TCR0 - Channel 0 Transfer Count Register

The Transfer Count Register (TCR) contains the number of bytes it transfers. This register is loaded when the transfer count field of the chain descriptor is read from memory. The maximum allowed value for the TCR is 16 Mbytes. Values greater than 16 Mbytes are truncated to 16 Mbytes and no error is reported. A value of zero is valid and results in no data being transferred and no cycles are generated on the source or destination buses and an interrupt is generated if enabled. During transfers, this register contains the remaining byte bytes to be written to the destination.

Table 16-307.Offset 28h: TCR0 - Channel 0 Transfer Count Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 28h **Offset End:** 2Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		0000000h	
23 : 00	TCR0	**Transfer Count:** Set by the IMCH when the transfer count field of the chain descriptor is read from memory. It reflects the number of bytes for a DMA transfer. A value of 0 results in no data being transferred. The maximum value that can be programmed to this register is 16 Mbytes. Larger values written in the transfer count field of the chain descriptor are truncated, and no error is reported. Refer to DCR0[18:17] for additional programming requirements when in Constant Address Mode.		000000h	RO

16.6.1.12 Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register

The Descriptor Control Register (DCR) contains control values for the DMA transfer on a per descriptor basis. This register is loaded when the descriptor control field of the chain descriptor is read from memory. The value for this register may vary from chain descriptor to chain descriptor.

Table 16-308.Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	TC	PCI Express A-segment Traffic Class: This field is used to set the Traffic Class field on the PCI Express bus for transactions. This field has no effect on memory to memory transactions.		000b	RO
28 : 19	Reserved	Reserved		000h	
18 : 17	Granularity	**Destination Granularity:** 00 1 byte granularity 01 2 byte granularity (DAR[0] and TCR[0] ignored) 10 4 byte granularity (DAR[1:0] and TCR[1:0] ignored) 11 Reserved These bits are loaded by the descriptor fetch only. This field is ignored unless bits 15:14 are 01b (selecting Constant Destination Mode). When this field is enabled, the Destination Address and Transfer Count Register must contain an integer multiple of the granularity.		00b	RO
16	Reserved	**Reserved**		0b	
15 : 14	DADDM	**Destination Processing Mode:** 00 Increment mode 01 Constant mode (bits 18:17 set the granularity) 10 Reserved 11 Reserved These bits are loaded by the descriptor fetch only.		00b	RO
13 : 12	SADDM	**Source Processing Mode:** 00 Increment mode 01 Decrement mode 10 Buffer Initialization 11 Reserved These bits are loaded by the descriptor fetch only.		00b	RO
11 : 09	Reserved	**Reserved**		000b	
08	SRCC	**Source Coherency:** 0 = Source is a non-coherent address space 1 = Source is a coherent address space		0b	RO
07	DSTC	**Destination Coherency:** 0 = Destination is a non-coherent address space 1 = Destination is a coherent address space For PCI-E writes (I/O), this bit inversely reflects the state of the Snoop Not Required Attribute header bit: 0 = Snoop not required attribute bit = 1 1 = Snoop not required attribute bit = 0		0b	RO

Table 16-308.Offset 2Ch: DCR0 - Channel 0 Descriptor Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	SRCT	**Source Type:** **Hardwired to 0:** 0 = Indicates the source address points to local system memory. 1 = Indicates that the source address points to I/O memory. Although an I/O source type is not supported, this bit will reflect the actual source type as written by software. The IMCH flags an error if the source address range is not valid, and ignores the value of this bit.		0b	RO
05	DSTT	**Destination Type:** 0 = Destination Address points to local system memory 1 = Destination Address points to I/O space An error is signaled by the IMCHif the Destination Address type read from the descriptor does not match this setting.		0b	RO
04	DABRTIE	**Aborted Interrupt Enable:** Indicates whether or not an interrupt is generated when the DMA Aborted (DABRT) bit in the CSR is set. 0 = Disable 1 = Enable		0b	RO
03	DSTPIE	**Stopped Interrupt Enable:** Indicates whether or not an interrupt is generated when the DMA Stopped (DSTP) bit in the CSR is set. 0 = Disable 1 = Enable		0b	RO
02	DSUSIE	**Suspended Interrupt Enable:** Indicates whether or not an interrupt is generated when the DMA Suspended (DSUS) bit in the CSR is set. 0 = Disable 1 = Enable		0b	RO
01	EOTIE	**End of Transfer Interrupt Enable:** Indicates whether or not an interrupt is generated when the End of Transfer (EOT) bit in the CSR is set. 0 = Disable 1 = Enable		0b	RO
00	EOCIE	**End of Chain Interrupt Enable:** Indicates whether or not an interrupt is generated when the End of Chain (EOC) bit in the CSR is set. 0 = Disable 1 = Enable		0b	RO

16.6.1.13 Offset 40h: CCR1 - Channel 1 Channel Control Register

Table 16-309.Offset 40h: CCR1 - Channel 1 Channel Control Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for CCR0 in Section 16.6.1.1.					

16.6.1.14 Offset 44h: CSR1 - Channel 1 Channel Status Register

Table 16-310.Offset 44h: CSR1 - Channel 1 Channel Status Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 44h **Offset End:** 47h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for CSR0 in Section 16.6.1.2.					

16.6.1.15 Offset 48h: CDAR1 - Channel 1 Current Descriptor Address Register

Table 16-311.Offset 48h: CDAR1 - Channel 1 Current Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 48h **Offset End:** 4Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for CDAR0 in Section 16.6.1.3.					

16.6.1.16 Offset 4Ch: CDUAR1 - Channel 1 Current Descriptor Upper Address Register

Table 16-312.Offset 4Ch: CDUAR1 - Channel 1 Current Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 4Ch **Offset End:** 4Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for CDUAR0 in Section 16.6.1.4.					

16.6.1.17 Offset 50h: SAR1 - Channel 1 Source Address Register

Table 16-313.Offset 50h: SAR1 - Channel 1 Source Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 50h **Offset End:** 53h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for SAR0 in Section 16.6.1.5.					

16.6.1.18 Offset 54h: SUAR1 - Channel 1 Source Upper Address Register

Table 16-314.Offset 54h: SUAR1 - Channel 1 Source Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 54h **Offset End:** 57h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for SUAR0 described in Section 16.6.1.6.					

16.6.1.19 Offset 58h: DAR1 - Channel 1 Destination Address Register

Table 16-315.Offset 58h: DAR1 - Channel 1 Destination Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 58h **Offset End:** 5Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those described for DAR0 in Section 16.6.1.7.					

16.6.1.20 Offset 5Ch: DUAR1 - Channel 1 Destination Upper Address Register

Table 16-316.Offset 5Ch: DUAR1 - Channel 1 Destination Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 5Ch **Offset End:** 5Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DUAR0 described in Section 16.6.1.8.					

16.6.1.21 Offset 60h: NDAR1 - Channel 1 Next Descriptor Address Register

Table 16-317.Offset 60h: NDAR1 - Channel 1 Next Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 60h **Offset End:** 63h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDAR0 described in Section 16.6.1.9.					

16.6.1.22 Offset 64h: NDUAR1 - Channel 1 Next Descriptor Upper Address Register

Table 16-318.Offset 64h: NDUAR1 - Channel 1 Next Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 64h **Offset End:** 67h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDUAR0 described in Section 16.6.1.10.					

16.6.1.23 Offset 68h: TCR1 - Channel 1 Transfer Count Register

Table 16-319.Offset 68h: TCR1 - Channel 1 Transfer Count Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for TCR0 described in Section 16.6.1.11.					

16.6.1.24 Offset 6Ch: DCR1 - Channel 1 Descriptor Control Register

Table 16-320.Offset 6Ch: DCR1 - Channel 1 Descriptor Control Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 6Ch **Offset End:** 6Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DCR0 described in Section 16.6.1.12.					

16.6.1.25 Offset 80h: CCR2 - Channel 2 Channel Control Register

Table 16-321.Offset 80h: CCR2 - Channel 2 Channel Control Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 80h **Offset End:** 83h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CCR0 described in Section 16.6.1.1.					

16.6.1.26 Offset 84h: CSR2 - Channel 2 Channel Status Register

Table 16-322.Offset 84h: CSR2 - Channel 2 Channel Status Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CSR0 described in Section 16.6.1.2.					

16.6.1.27 Offset 88h: CDAR2 - Channel 2 Current Descriptor Address Register

Table 16-323.Offset 88h: CDAR2: Channel 2 Current Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 88h **Offset End:** 8Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CDAR0 described in Section 16.6.1.3.					

16.6.1.28 Offset 8Ch: CDUAR2 - Channel 2 Current Descriptor Upper Address Register

Table 16-324.Offset 8Ch: CDUAR2 - Channel 2 Current Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 8Ch **Offset End:** 8Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CDUAR0 described in Section 16.6.1.4.					

16.6.1.29 Offset 90h: SAR2 - Channel 2 Source Address Register

Table 16-325.Offset 90h: SAR2 - Channel 2 Source Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 90h **Offset End:** 93h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for SAR0 described in Section 16.6.1.5.					

16.6.1.30 Offset 94h: SUAR2 - Channel 2 Source Upper Address Register

Table 16-326.Offset 94h: SUAR2 - Channel 2 Source Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 94h **Offset End:** 97h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for SUAR0 described in Section 16.6.1.6.					

16.6.1.31 Offset 98h: DAR2 - Channel 2 Destination Address Register

Table 16-327.Offset 98h: DAR2 - Channel 2 Destination Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 98h **Offset End:** 9Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DAR0 described in Section 16.6.1.7.					

16.6.1.32 Offset 9Ch: DUAR2 - Channel 2 Destination Upper Address Register

Table 16-328.Offset 9Ch: DUAR2 - Channel 2 Destination Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** 9Ch **Offset End:** 9Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DUAR0 described in Section 16.6.1.8.					

16.6.1.33 Offset A0h: NDAR2 - Channel 2 Next Descriptor Address Register

Table 16-329.Offset A0h: NDAR2 - Channel 2 Next Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDAR0 described in Section 16.6.1.9.					

16.6.1.34 Offset A4h: NDUAR2 - Channel 2 Next Descriptor Upper Address Register

Table 16-330.Offset A4h: NDUAR2 - Channel 2 Next Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** A4h **Offset End:** A7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDUAR0 described in Section 16.6.1.10.					

16.6.1.35 Offset A8h: TCR2 - Channel 2 Transfer Count Register

Table 16-331.Offset A8h: DCR2 - Channel 2Transfer Control Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** A8h **Offset End:** ABh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DCR0 described in Section 16.6.1.11.					

16.6.1.36 Offset ACh: DCR2 - Channel 2 Descriptor Control Register

Table 16-332.Offset ACh: DCR2 - Channel 2 Descriptor Control Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** ACh **Offset End:** AFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DCR0 described in Section 16.6.1.12.					

16.6.1.37 Offset C0h: CCR3 - Channel 3 Channel Control Register

Table 16-333.Offset C0h: CCR3 - Channel 3 Channel Control Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** C0h **Offset End:** C3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CCR0 described in Section 16.6.1.1.					

16.6.1.38 Offset C4h: CSR3 - Channel 3 Channel Status Register

Table 16-334.Offset C4h: CSR3 - Channel 3 Channel Status Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** C4h **Offset End:** C7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CSR0 described in Section 16.6.1.2.					

16.6.1.39 Offset C8h: CDAR3 - Channel 3 Current Descriptor Address Register

Table 16-335.Offset C8h: CDAR3 - Channel 3 Current Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** C8h **Offset End:** CBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CDAR0 described in Section 16.6.1.3.					

16.6.1.40 Offset CCh: CDUAR3 - Channel 3 Current Descriptor Upper Address Register

Table 16-336.Offset CCh: CDUAR3 - Channel 3 Current Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** CCh **Offset End:** CFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for CDUAR0 described in Section 16.6.1.4.					

16.6.1.41 Offset D0h: SAR3 - Channel 3 Source Address Register

Table 16-337.Offset D0h: SAR3 - Channel 3 Source Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** D0h **Offset End:** D3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for SAR0 described in Section 16.6.1.5.					

16.6.1.42 Offset D4h: SUAR3 - Channel 3 Source Upper Address Register

Table 16-338.Offset D4h: SUAR3 - Channel 3 Source Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** D4h **Offset End:** D7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for SUAR0 described in Section 16.6.1.6.					

16.6.1.43 Offset D8h: DAR3 - Channel 3 Destination Address Register

Table 16-339.Offset D8h: DAR3 - Channel 3 Destination Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** D8h **Offset End:** DBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DAR0 described in Section 16.6.1.7.					

16.6.1.44 Offset DCh: DUAR3 - Channel 3 Destination Upper Address Register

Table 16-340.Offset DCh: DUAR3 - Channel 3 Destination Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** DCh **Offset End:** DFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DUAR0 described in Section 16.6.1.8.					

16.6.1.45 Offset E0h: NDAR3 - Channel 3 Next Descriptor Address Register

Table 16-341.Offset E0h: NDAR3 - Channel 3 Next Descriptor Address Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** E0h **Offset End:** E3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDAR0 described in Section 16.6.1.9.					

16.6.1.46 Offset E4h: NDUAR3 - Channel 3 Next Descriptor Upper Address Register

Table 16-342.Offset E4h: NDUAR3 - Channel 3 Next Descriptor Upper Address Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** E4h **Offset End:** E7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for NDUAR0 described in Section 16.6.1.10.					

16.6.1.47 Offset E8h: TCR3 - Channel 3 Transfer Count Register

Table 16-343.Offset E8h: TCR3 - Channel 3 Transfer Count Register

Description:					
View: PCI	**BAR:** EDMALBAR		**Bus:Device:Function:** 0:1:0	**Offset Start:** E8h **Offset End:** EBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for TCR0 described in SSection 16.6.1.11.					

16.6.1.48 Offset ECh: DCR3 - Channel 3 Descriptor Control Register

Table 16-344.Offset ECh: DCR3 - Channel 3 Descriptor Control Register

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** ECh **Offset End:** EFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
The bit descriptions for this register are identical to those for DCR0 described in Section 16.6.1.12.					

16.6.1.49 Offset 100h: DCGC - EDMA Controller Global Command

This register controls enabling and designation of priority channel.

Table 16-345.Offset 100h: DCGC - EDMA Controller Global Command

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 100h **Offset End:** 103h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 03	Reserved	Reserved		0	
02	PCENBL	**Priority Channel Enable:** 0 = No priority channel. The Priority Channel Select bits are ignored. 1 = Enable the Priority Channel as programmed by the Priority Channel Select.		0b	RW
01 : 00	PCSLT	**Priority Channel Selects:** When Priority Channel Enable is set, the DMA channel selected by this field has a higher priority than the others. 00 Channel 0 is the Priority Channel 01 Channel 1 is the Priority Channel 10 Channel 2 is the Priority Channel 11 Channel 3 is the Priority Channel		00b	RW

16.6.1.50 Offset 104h: DCGS - EDMA Controller Global Status

This register is accessed by the device driver to determine the source of an interrupt from the EDMA controller.

Table 16-346.Offset 104h: DCGS - EDMA Controller Global Status

Description:					
View: PCI	**BAR:** EDMALBAR	**Bus:Device:Function:** 0:1:0		**Offset Start:** 104h **Offset End:** 107h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 26	Reserved	Reserved		00h	
25	NIC3	**Normal Interrupt Condition from Channel 3:** 0 = No Channel 3 Normal Interrupt is generated. 1 = A Channel 3 Normal Interrupt has been generated.		0b	RO
24	EIC3	**Error interrupt Condition from Channel 3:** 0 = No Channel 3 Error Interrupt is generated. 1 = A Channel 3 Error Interrupt has been generated. The channel is in Abort status.		0b	RO
23 : 18	Reserved	Reserved		00h	
17	NIC2	**Normal Interrupt Condition from Channel 2:** 0 = No Channel 2 Normal Interrupt is generated. 1 = A Channel 2 Normal Interrupt has been generated.		0b	RO
16	EIC2	**Error interrupt Condition from Channel 2:** 0 = No Channel 2 Error Interrupt is generated. 1 = A Channel 2 Error Interrupt has been generated. The channel is in Abort status.		0b	RO
15 : 10	Reserved	Reserved		00h	
09	NIC1	**Normal Interrupt Condition from Channel 1:** 0 = No Channel 1 Normal Interrupt is generated. 1 = A Channel 1 Normal Interrupt has been generated.		0b	RO
08	EIC1	**Error interrupt Condition from Channel 1:** 0 = No Channel 1 Error Interrupt is generated. 1 = A Channel 1 Error Interrupt has been generated. The channel is in Abort status.		0b	RO
07 : 02	Reserved	Reserved		00h	
01	NIC0	**Normal Interrupt Condition from Channel 0:** 0 = No Channel 0 Normal Interrupt is generated. 1 = A Channel 0 Normal Interrupt has been generated.		0b	RO
00	EIC0	**Error interrupt Condition from Channel 0:** 0 = No Channel 0 Error Interrupt is generated. 1 = A Channel 0 Error Interrupt has been generated. The channel is in Abort status.		0b	RO

16.7 Memory Mapped I/O for NSI Registers

This section describes the memory-mapped registers for the North South Interface (NSI). The NSIBAR register, described in Section 16.1, "IMCH Registers: Bus 0, Device 0, Function 0" provides the base address for these registers. The offsets listed for the following registers are relative to this base address.

This Root Complex Register Block (RCRB) controls CMI's internal serial interconnect bus..

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 16-347.Bus 0, Device 0, Function 0: Summary of IMCH Configuration Registers Mapped Through NSIBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: SNSIVCECH - NSI Virtual Channel Enhanced Capability Header Register" on page 680	04010002h
04h	07h	"Offset 04h: NSIPVCCAP1 - NSI Port VC Capability Register 1" on page 680	00000000h
08h	0Bh	"Offset 08h: NSIPVCCAP2 - Port VC Capability Register 2" on page 681	00000001h
0Ch	0Dh	"Offset 0Ch: NSIPVCCTL - NSI Port VC Control Register" on page 682	0000h
10h	13h	"Offset 10h: NSIVC0RCAP - NSI VC0 Resource Capability Register" on page 682	00000001h
14h	17h	"Offset 14h: NSIVC0RCTL - NSI VC0 Resource Control Register" on page 683	800000FFh
1Ah	1Bh	"Offset 1Ah: NSIVC0RSTS - NSI VC0 Resource Status Register" on page 684	0002h
80h	83h	"Offset 80h: NSIRCILCECH - NSI Root Complex Internal Link Control Enhanced Capability Header Register" on page 684	00010006h
84h	87h	"Offset 84h: NSILCAP - NSI Link Capabilities Register" on page 685	0003A041h

16.7.1 Register Details

16.7.1.1 Offset 00h: NSIVCECH - NSI Virtual Channel Enhanced Capability Header Register

This register indicates NSI Virtual Channel capabilities.

Table 16-348.Offset 00h: SNSIVCECH - NSI Virtual Channel Enhanced Capability Header Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** 04010002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	PNC	**Pointer to Next Capability:** This field contains the offset to the next PCI Express capability structure in the linked list of capabilities (Link Declaration Capability). Bits [21:20] are reserved and software must mask them to allow for future uses of these bits.		040h	RO
19 : 16	PCIEVCC	**PCI Express Virtual Channel Capability Version:** Hardwired to 1 to indicate compliances with the 1.0 version of the PCI Express Specification.		1h	RO
15 : 00	ECID	**Extended Capability ID:** Value of 0002h identifies this linked list item (capability structure) as being for PCI Express Virtual Channel registers.		0002h	RO

16.7.1.2 Offset 04h: NSIPVCCAP1 - NSI Port VC Capability Register 1

This register describes the configuration of PCI Express Virtual Channels associated with this port.

Table 16-349.Offset 04h: NSIPVCCAP1 - NSI Port VC Capability Register 1 (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 07	Reserved	Reserved		0000000h	

Table 16-349.Offset 04h: NSIPVCCAP1 - NSI Port VC Capability Register 1 (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** NSIBAR		**Bus:Device:Function:** 0:0:0	**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06 : 04	LPEVCC	**Low Priority Extended VC Count:** Indicates the number of (extended) Virtual Channels in addition to the default VC belonging to the low-priority VC (LPVC) group that has the lowest priority with respect to other VC resources in a strict-priority VC Arbitration. The value of 0 in this field implies strict VC arbitration.		000b	RO
03	Reserved	Reserved		0b	
02 : 00	EVCC	**Extended VC Count:** Indicates the number of (extended) Virtual Channels in addition to the default VC supported by the device. The Private Virtual Channel is not included in this count. Only VC0 is supported.		000b	RO

16.7.1.3 Offset 08h: NSIPVCCAP2 - Port VC Capability Register 2

This register describes the configuration of PCI Express Virtual Channels associated with this port.

Table 16-350.Offset 08h: NSIPVCCAP2 - Port VC Capability Register 2

Description:					
View: PCI	**BAR:** NSIBAR		**Bus:Device:Function:** 0:0:0	**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		00h	
23 : 08	Reserved	Reserved		0000h	
07 : 00	VCARBC	**VC Arbitration Capability:** Indicates that the only possible VC arbitration scheme is hardware fixed (in the root complex).		01h	RO

16.7.1.4 Offset 0Ch: NSIPVCCTL - NSI Port VC Control Register

Table 16-351.Offset 0Ch: NSIPVCCTL - NSI Port VC Control Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 0Ch **Offset End:** 0Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	Reserved	Reserved		000h	
03 : 01	VCARBSEL	**VC Arbitration Select:** This field is programmed by software to the only possible value as indicated in the VC Arbitration Capability field. The value 000b when written to this field indicates the VC arbitration scheme is hardware fixed (in the root complex). This field cannot be modified when more than one VC in the LPVC group is enabled.		000b	RW
00	Reserved	Reserved		0b	

16.7.1.5 Offset 10h: NSIVC0RCAP - NSI VC0 Resource Capability Register

Table 16-352.Offset 10h: NSIVC0RCAP - NSI VC0 Resource Capability Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		00h	
15	RSNPT	**Reject Snoop Transactions:** 0 = Transactions with or without the No Snoop bit set within the TLP header are allowed on this VC. 1 = Any transaction without the No Snoop bit set within the TLP header is rejected as an Unsupported Request.		0b	RO
14 : 08	Reserved	Reserved		00h	
07 : 00	PARBC	**Port Arbitration Capability:** Having only bit 0 set indicates that the only supported arbitration scheme for this VC is non-configurable hardware-fixed.		01h	RO

16.7.1.6 Offset 14h: NSIVC0RCTL - NSI VC0 Resource Control Register

This register controls the resources associated with PCI Express Virtual Channel 0.

Table 16-353.Offset 14h: NSIVC0RCTL - NSI VC0 Resource Control Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 800000FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	VC0EN	**VC0 Enable:** Hardwired to 1. VC0 can never be disabled.		1b	RO
30 : 27	Reserved	Reserved		0h	
26 : 24	VC0ID	**VC0 ID:** Assigns a VC ID to the VC resource. For VC0 this is hardwired to 0 and read only.		000b	RO
23 : 20	Reserved	Reserved		0h	
19 : 17	PARBSEL	**Port Arbitration Select:** Configures the VC resource to provide a particular Port Arbitration service. Valid value for this field is a number corresponding to one of the asserted bits in the Port Arbitration Capability field of the VC resource. Because only bit 0 of that field is asserted. This field is always programmed to '1'.		0h	RW
16 : 08	Reserved	Reserved		000h	
07 : 01	TCVC0M	**TC/VC0 Map:** Indicates the TCs (Traffic Classes) that are mapped to the VC resource. Bit locations within this field correspond to TC values. For example, when bit 7 is set in this field, TC7 is mapped to this VC resource. When more than one bit in this field is set, it indicates that multiple TCs are mapped to the VC resource. In order to remove one or more TCs from the TC/VC Map of an enabled VC, software must ensure that no new or outstanding transactions with the TC labels are targeted at the given Link.		7Fh	RW
00	TC0VC0M	**TC0/VC0 Map:** Traffic Class 0 is always routed to VC0. Thus is will always read as 1b.		1b	RO

16.7.1.7 Offset 1Ah: NSIVC0RSTS - NSI VC0 Resource Status Register

This register reports the Virtual Channel specific status.

Table 16-354.Offset 1Ah: NSIVC0RSTS - NSI VC0 Resource Status Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 1Ah **Offset End:** 1Bh	
Size: 16 bit	**Default:** 0002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 02	Reserved	Reserved		0000h	
01	VC0NP	**VC0 Negotiation Pending:** 0 = The VC negotiation is complete. 1 = The VC resource is still in the process of negotiation (initialization or disabling). This bit indicates the status of the process of flow control initialization. It is set by default on reset, as well as whenever the corresponding virtual channel is disabled or the link is in the DL_Down state. It is cleared when the link successfully exits the FC_INIT2 state. Before using a virtual channel, software must check whether the VC negotiation pending fields for that virtual channel are cleared in both components on link.		1b	RO
00	Reserved	Reserved		0b	

16.7.1.8 Offset 80h: NSIRCILCECH - NSI Root Complex Internal Link Control Enhanced Capability Header Register

This capability contains controls for the Root Complex Internal Link known as NSI.

Table 16-355.Offset 80h: NSIRCILCECH - NSI Root Complex Internal Link Control Enhanced Capability Header Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 80h **Offset End:** 83h	
Size: 32 bit	**Default:** 00010006h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	PNTNC	**Pointer to Next Capability:** This value terminates the PCI Express extended capabilities list associated with this RCRB.		000h	RO
19 : 16	LNKDC	**Link Declaration Capability Version:** Hardwired to 1 to indicate compliances with the 1.0 version of the PCI Express specification.		1h	RO
15 : 00	EXTCID	**Extended Capability ID:** Value of 0006h identifies this linked list item (capability structure) as being for PCI Express Internal Link Control Capability.		0006h	RO

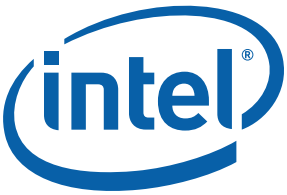
16.7.1.9 Offset 84h: NSILCAP - NSI Link Capabilities Register

This register indicates NSI specific capabilities.

Table 16-356.Offset 84h: NSILCAP - NSI Link Capabilities Register

Description:					
View: PCI	**BAR:** NSIBAR	**Bus:Device:Function:** 0:0:0		**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 0003A041h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 18	Reserved	Reserved		00h	
17 : 15	L1ELAT	**L1 Exit Latency:** Indicates the length of time this Port requires to complete the transition from L1 to L0. The value 010 b indicate more than 64 µs. Both bytes of this register that contain a portion of this field must be written simultaneously in order to prevent an intermediate (and undesired) value from ever existing.		111b	RWO
14 : 12	Reserved	Reserved		010b	
11 : 10	ASLPMS	**Active State Link PM Support: Reserved**		00b	
09 : 04	MXLW	**Maximum Link Width:** Hardwired to indicate X4.		04h	RO
03 : 00	MXLS	**Maximum Link Speed:** Hardwired to indicate 2.5 Gbytes/s.		1h	RO

§ §



Integrated I/O Controller Hub, Volume 3 of 6

Order Number: 320066-003US

17.0 Bridging and Configuration

17.1 Root Complex Memory-Mapped Configuration Register Details

This section describes all registers and base functionality that are related to configuration and not a specific interface (such as LPC). It contains the root complex register block, which describes the behavior of the upstream internal link.

The root complex register block is mapped into memory space using register RCBA (see Section 19.2.7.1, "Offset F0h: RCBA: Root Complex Base Address Register"). Accesses in this space must be limited to 32-bit (Dword) quantities. Burst accesses are not allowed.

Note: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failures or undetermined behavior. For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables").

Table 17-1. Bus 0, Device 31, Function 0: Summary of Root Complex Configuration Registers Mapped Through RCBA Memory BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: VCH - Virtual Channel Capability Header Register" on page 691	10010002h
0004h	0007h	"Offset 0004h: VCAP1 - Virtual Channel Capability 1 Register" on page 691	0801h
0008h	000Bh	"Offset 0008h: VCAP2 - Virtual Channel Capability 2 Register" on page 692	0001h
000Ch	000Dh	"Offset 000Ch: PVC - Port Virtual Channel Control Register" on page 692	0h
000Eh	000Fh	"Offset 000Eh: PVS -Port Virtual Channel Status Register" on page 693	0h
0010h	0013h	"Offset 0010h: V0CAP - Virtual Channel 0 Resource Capability Register" on page 693	00000001h
0014h	0017h	"Offset 0014h: V0CTL - Virtual Channel 0 Resource Control Register" on page 694	800000FFh
001Ah	001Bh	"Offset 001Ah: V0STS - Virtual Channel 0 Resource Status Register" on page 695	0h
0100h	0103h	"Offset 0100h: RCTCL - Root Complex Topology Capabilities List Register" on page 696	1A010005h
0104h	0107h	"Offset 0104h: ESD - Element Self Description Register" on page 696	00000102h
0110h	0113h	"Offset 0110h: ULD - Upstream Link Description Register" on page 697	0001h
0118h	011Fh	"Offset 0118h: ULBA - Upstream Link Base Address Register" on page 697	0000000000000000h
01A0h	01A3h	"Offset 01A0h: ILCL - Internal Link Capabilities List Register" on page 698	00010006h
01A4h	01A7h	"Offset 01A4h: LCAP - Link Capabilities Register" on page 698	0012441h
01A8h	01A9h	"Offset 01A8h: LCTL - Link Control Register" on page 699	0h

Table 17-1. Bus 0, Device 31, Function 0: Summary of Root Complex Configuration Registers Mapped Through RCBA Memory BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
01AAh	01ABh	"Offset 01AAh: LSTS - Link Status Register" on page 700	0041h
3108h	310Bh	"Offset 3108h: D29IP - Device 29 Interrupt Pin Register" on page 702	10004321h
3140h	3141h	"Offset 3140h: D31IR - Device 31 Interrupt Route Register" on page 702	3210h
3144h	3145h	"Offset 3144h: D29IR - Device 29 Interrupt Route Register" on page 703	3210h
31FFh	31FFh	"Offset 31FFh: OIC - Other Interrupt Control Register" on page 704	0h
3400h	3403h	"Offset 3400h: RC - RTC Configuration Register" on page 704	0h
3404h	3407h	"Offset 3404h: HPTC - High Performance Precision Timer Configuration Register" on page 705	0h
3410h	3413h	"Offset 3410h: GCS - General Control and Status Register" on page 706	Variable
3414h	3417h	"Offset 3414h: BUC - Backed Up Control Register" on page 708	Variable
3418h	341Bh	"Offset 3418h: FD - Function Disable Register" on page 709	00000080h
341Ch	341Fh	"Offset 341Ch: PRC - Power Reduction Control Register Clock Gating" on page 711	0h

Table 17-2. RCBA Base Address Registers in the IA F View

Offset Start	Offset End	Register ID - Description	Default Value
3000h	3001h	"Offset 3000h: TCTL - TCO Control Register" on page 700	0h
3100h	3103h	"Offset 3100h: D31IP - Device 31 Interrupt Pin Register" on page 701	00042210h

17.1.1 VC Configuration Registers

17.1.1.1 Offset 0000h: VCH - Virtual Channel Capability Header Register

Table 17-3. Offset 0000h: VCH - Virtual Channel Capability Header Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0000h **Offset End:** 0003h	
Size: 32 bit	**Default:** 10010002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :20	NCO	**Next Capability Offset:** Indicates the next item in the list.		100h	RO
19 :16	CV	**Capability Version:** Indicates this is version 1 of the capability structure by the PCI SIG.		1h	RO
15 :00	CID	**Capability ID:** Indicates this is the Virtual Channel capability item.		0002h	RO

17.1.1.2 Offset 0004h: VCAP1 - Virtual Channel Capability 1 Register

Table 17-4. Offset 0004h: VCAP1 - Virtual Channel Capability 1 Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0004h **Offset End:** 0007h	
Size: 32 bit	**Default:** 0801h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :12	Reserved	Reserved		0h	
11 :10	Reserved	Reserved		10b	
09 :08	RC	**Reference Clock:** Fixed at 100 ns for this version of the *PCI Express Specification*.		00b	RO
07	Reserved	Reserved		0h	
06 :04	LPEVC	**Low Priority Extended VC Count:** Indicates that there are no additional VCs of low priority with extended capabilities.		0h	RO
03	Reserved	Reserved		0h	
02 :00	Reserved	Reserved.		001h	

17.1.1.3 Offset 0008h: VCAP2 - Virtual Channel Capability 2 Register

Table 17-5. Offset 0008h: VCAP2 - Virtual Channel Capability 2 Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0008h **Offset End:** 000Bh	
Size: 32bit	**Default:** 0001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	ATO	**VC Arbitration Table Offset:** Indicates that no table is present for VC arbitration since it is fixed.		00h	RO
23 :08	Reserved	Reserved		0h	
07 :00	AC	**VC Arbitration Capability:** Indicates that the VC arbitration is fixed in the root complex.		01h	RO

17.1.1.4 Offset 000Ch: PVC - Port Virtual Channel Control Register

Table 17-6. Offset 000Ch: PVC - Port Virtual Channel Control Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 000Ch **Offset End:** 000Dh	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :04	Reserved	Reserved		000h	
03 :01	AS	**VC Arbitration Select:** Indicates which VC must be programmed in the VC arbitration table. The root complex takes no action on the setting of this field since there is no arbitration table.		0h	RW
00	LAT	**Load VC Arbitration Table:** Indicates that the table programmed must be loaded into the VC arbitration table. This bit is defined as read/write with always returning 0 on reads. Since there is no VC arbitration table in the root complex, this bit can be built as read-only.		0h	RO

17.1.1.5 Offset 000Eh: PVS - Port Virtual Channel Status Register

Table 17-7. Offset 000Eh: PVS -Port Virtual Channel Status Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 000Eh **Offset End:** 000Fh	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :01	Reserved	Reserved		0000h	
00	VAS	**VC Arbitration Table Status:** Indicates the coherency status of the VC Arbitration table when it is being updated. This field is hardwired to 0 in the root complex since there is no VC arbitration table.		0h	RO

17.1.1.6 Offset 0010h: V0CAP - Virtual Channel 0 Resource Capability Register

Table 17-8. Offset 0010h: V0CAP - Virtual Channel 0 Resource Capability Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0010h **Offset End:** 0013h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	AT	**Port Arbitration Table Offset:** This VC implements no port arbitration table since the arbitration is fixed.		00h	RO
23	Reserved	Reserved		0h	
22 :16	MTS	**Maximum Time Slots:** This VC implements fixed arbitration, and therefore this field is not used.		00h	RO
15	RTS	**Reject Snoop Transactions:** This VC must be able to take snoopable transactions.		0h	RO
14	APS	**Advanced Packet Switching:** This VC is capable of all transactions, not just advanced packet switching transactions.		0h	RO
13 :08	Reserved	Reserved		0h	
07 :00	PAC	**Port Arbitration Capability:** Indicates that this VC uses fixed port arbitration.		01h	RO

17.1.1.7 Offset 0014h: V0CTL - Virtual Channel 0 Resource Control Register

Table 17-9. Offset 0014h: V0CTL - Virtual Channel 0 Resource Control Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0014h **Offset End:** 0017h	
Size: 32 bit	**Default:** 800000FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	EN	**Virtual Channel Enable:** Enables the VC when set. Disables the VC when cleared. 0 = Disables the VC 1 = Enables the VC		1h	RO
30 :27	Reserved	Reserved		0h	
26 :24	ID	**Virtual Channel Identifier:** Indicates the ID to use for this virtual channel.		0h	RO
23 :20	Reserved	Reserved		0h	
19 :17	PAS	**Port Arbitration Select:** Indicates which port table is being programmed. The root complex takes no action on this setting since the arbitration is fixed and there is no arbitration table.		0h	RW
16	LAT	**Load Port Arbitration Table:** The root complex does not implement an arbitration table for this virtual channel.		0h	RO
15 :08	Reserved	Reserved		00h	
07 :01	TCVC0M	**Transaction Class / Virtual Channel 0 Map:** Indicates which transaction classes are mapped to this virtual channel. 0 = This transaction class is not mapped to the virtual channel 0. 1 = This transaction class is mapped to the virtual channel 0.		7Fh	RW
00	TC0VC0M	**Transaction Class 0 / Virtual Channel 0 Map:** Indicates that transaction class 0 is always mapped to VC0.		1h	

17.1.1.8 Offset 001Ah: V0STS - Virtual Channel 0 Resource Status Register

Table 17-10. Offset 001Ah: V0STS - Virtual Channel 0 Resource Status Register

Description:						
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0		**Offset Start:** 001Ah **Offset End:** 001Bh	
Size: 16 bit	**Default:** 0h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15 :02	Reserved	Reserved			0000h	
01	NP	**VC Negotiation Pending:** 0 = Indicates the virtual channel is not being negotiated with ingress ports. 1 = Indicates the virtual channel is still being negotiated with ingress ports.			0h	RO
00	ATS	**Port Arbitration Table Status:** There is no port arbitration table for this VC so this bit is reserved at 0.			0h	RO

17.1.2 Root Complex Topology Capability Structure Registers

The following registers follow the PCI Express capability list structure as defined in the *PCI Express* Specification*, to indicate the capabilities of NSI.

17.1.2.1 Offset 0100h: RCTCL- - Root Complex Topology Capabilities List Register

Table 17-11. Offset 0100h: RCTCL - Root Complex Topology Capabilities List Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0100h **Offset End:** 0103h	
Size: 32 bit	**Default:** 1A010005h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :20	NEXT	**Next Capability:** Indicates next item in the list.		1A0h	RO
19 :16	CV	**Capability Version:** Indicates the version of the capability structure.		1h	RO
15 :00	CID	**Capability ID:** Indicates this is a PCI Express link capability section of an RCRB.		0005h	RO

17.1.2.2 Offset 0104h: ES - Element Self Description Register

Table 17-12. Offset 0104h: ESD - Element Self Description Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0104h **Offset End:** 0107h	
Size: 32 bit	**Default:** 00000102h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	PN	**Port Number:** A value of 0 to indicate the egress port for the IICH.		00h	RO
23 :16	CID	**Component ID:** Indicates the component ID assigned to this element by software. This is written once by platform BIOS and is locked until a platform reset.		00h	RWO
15 :08	NLE	**Number of Link Entries:** Indicates that one link entry (corresponding to NSI) is described by this RCRB.		01h	RO
07 :04	Reserved	Reserved		0h	
03 :00	ET	**Element Type:** Indicates that the element type is a root complex internal link.		2h	RO

17.1.2.3 Offset 0110h: ULD - Upstream Link Description Register

Table 17-13. Offset 0110h: ULD - Upstream Link Description Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0110h **Offset End:** 0113h	
Size: 32 bit	**Default:** 0001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	PN	**Target Port Number:** This field is programmed by platform BIOS to match the port number of the IMCH. RCRB that is attached to this RCRB.		00h	RWO
23 :16	TCID	**Target Component ID:** This field is programmed by platform BIOS to match the component ID of the IMCH. RCRB that is attached to this RCRB.		00h	RWO
15 :02	Reserved	Reserved		0h	
01	LT	**Link Type:** Indicates that the link points to the IMCH RCRB.		0h	RO
00	LV	**Link Valid:** Indicates that this link entry is valid.		1h	RO

17.1.2.4 Offset 0118h: ULBA - Upstream Link Base Address Register

Table 17-14. Offset 0118h: ULBA - Upstream Link Base Address Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0118h **Offset End:** 011Fh	
Size: 64 bit	**Default:** 0000000000000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :32	BAU	**Base Address Upper:** This field is programmed by platform BIOS to match the upper 32-bits of base address of the IMCH. RCRB that is attached to this RCRB.		00000000h	RWO
31 :00	BAL	**Base Address Lower:** This field is programmed by platform BIOS to match the lower 32-bits of base address of the IMCH. RCRB that is attached to this RCRB.		00000000h	RWO

17.1.3 Internal Link Configuration Registers

17.1.3.1 Offset 01A0h: ILCL - Internal Link Capabilities List Register

Table 17-15. Offset 01A0h: ILCL - Internal Link Capabilities List Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 01A0h **Offset End:** 01A3h	
Size: 32 bit	**Default:** 00010006h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :20	NEXT	**Next Capability:** Indicates this is the last item in the list.		000h	RO
19 :16	CV	**Capability Version:** Indicates the version of the capability structure.		1h	RO
15 :00	CID	**Capability ID:** Indicates this is the capability for NSI.		0006h	RO

17.1.3.2 Offset 01A4h: LCAP - Link Capabilities Register

Table 17-16. Offset 01A4h: LCAP - Link Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 01A4h **Offset End:** 01A7h	
Size: 32 bit	**Default:** 0012441h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :18	Reserved	Reserved		0h	
17 :15	EL1	**The EP80579 does not support L0s or L1.** **L1 Exit Latency:** Indicates that the exit latency is 2 µs to 4 µs.		010b	RO
14 :12	EL0	**The EP80579 does not support L0s or L1.** **L0s Exit Latency:** This field is read/write and updatable by BIOS. It defaults to 128 ns to less than 256 ns, assuming a common-clock configuration between IICH and IMCH. If a unique clock value is used, it is recommended that BIOS update this field to 100 (512 ns to less than 1 µs). When BIOS set sets this field, it must also update NSI's DBG.NFTS field, located at offset 2024h in configuration space.		010b	RW

Table 17-16. Offset 01A4h: LCAP - Link Capabilities Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 01A4h **Offset End:** 01A7h	
Size: 32 bit	**Default:** 0012441h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 :10	APMS	**The EP80579 does not support L0s or L1.** **Active State Link PM Support:** Indicates the level of active state power management on NSI. Bits Definition 00 Neither L0s nor L1 supported 01 L0s Entry supported (Per PCI Express spec, L0s must be supported, but the EP80579 has defeatured L0s.) 10 Reserved: L1 Entry not supported on NSI 11 Reserved: L1 Entry not supported on NSI		1h	RWO
09 :04	MLW	**Maximum Link Width:** Indicates the maximum link width is four ports.		4h	RO
03 :00	MLS	**Maximum Link Speed:** Indicates the link speed is 2.5 Gbits/s.		1h	RO

17.1.3.3 Offset 01A8h: LCTL - Link Control Register

Table 17-17. Offset 01A8h: LCTL - Link Control Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 01A8h **Offset End:** 01A9h	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	Reserved	Reserved		00h	
07 :07		Reserved		0	
06 :02	Reserved	Reserved		0h	
01 :00	APMC	L0s has been defeatured on WL, and ASPM must never be turned on. **Active State Link PM Control:** Indicates whether NSI must enter L0s or L1 or both. Bits Definition 00 Disabled 01 L0s Entry Enabled 10 L1 Entry Enabled 11 L0s and L1 Entry Enabled		0h	RW

17.1.3.4 Offset 01AAh: LSTS - Link Status Register

Table 17-18. Offset 01AAh: LSTS - Link Status Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 01AAh **Offset End:** 01ABh	
Size: 16 bit	**Default:** 0041h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :10	Reserved	Reserved		00h	
09 :04	NLW	**Negotiated Link Width:** Minimum negotiated link width is a x4 port. The contents of this register are undefined if the link has not successfully trained.		4h	RO
03 :00	LS	**Link Speed:** Link is 2.5 Gbits/s.		1h	RO

17.1.4 TCO Configuration

17.1.4.1 Offset 3000h: TCTL - TCO Control Register

Table 17-19. Offset 3000h: TCTL - TCO Control Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** RCBA			**Offset Start:** 3000h **Offset End:** 3001h	
Size: 8 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	IE	**TCO IRQ Enable:** 0 = TCO IRQ is disabled. 1 = TCO IRQ is enabled, as selected by the TCO_IRQ_SEL field.		0h	RW

Table 17-19. Offset 3000h: TCTL - TCO Control Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** RCBA			**Offset Start:** 3000h **Offset End:** 3001h	
Size: 8 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06 : 03	Reserved	Reserved		0h	
02 : 00	IS	**TCO IRQ Select:** Specifies on which IRQ the TCO internally appears. If not using the APIC, the TCO interrupt must be routed to IRQ9-11, and that interrupt is not sharable with the SERIRQ stream, but it can be shared with other PCI interrupts. If using the APIC, the TCO interrupt can also be mapped to IRQ20-23 and can be shared with other interrupt. **Bits SCI Map** 000 IRQ9 001 IRQ10 010 IRQ11 011 Reserved 100 IRQ20 (only if APIC enabled) 101 IRQ21 (only if APIC enabled) 110 IRQ22 (only if APIC enabled) 111 IRQ23 (only if APIC enabled) When setting these bits, the IE bit must be cleared to prevent glitches. When the interrupt is mapped to APIC interrupts 9, 10, or 11, the APIC must be programmed for active-high reception. When the interrupt is mapped to APIC interrupts 20 through 23, the APIC must be programmed for active-low reception.		000h	RW

17.1.5 Interrupt Configuration Registers

17.1.5.1 Offset 3100h: D31IP - Device 31 Interrupt Pin Register

Table 17-20. Offset 3100h: D31IP - Device 31 Interrupt Pin Register

Description:					
View: IA F	**Base Address:** RCBA			**Offset Start:** 3100h **Offset End:** 3103h	
Size: 32 bit	**Default:** 00042210h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	
15 : 12	SMIP	**SM Bus Pin:** See the CIP description. This field applies to the SM Bus controller.		2h	RW
11 : 08	SIP	**SATA Pin:** See the CIP description. This field applies to the SATA controller.		2h	RW
07 : 04	Reserved	Reserved		1h	
03 : 00	PIP	**PCI Bridge Pin:** See the CIP description. This field applies to the PCI bridge. Currently, the PCI bridge does not generate an interrupt so this field is read-only and '0'.		0h	RO

17.1.5.2 Offset 3108h: D29IP - Device 29 Interrupt Pin Register

Table 17-21. Offset 3108h: D29IP - Device 29 Interrupt Pin Register

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 3108h **Offset End:** 310Bh	
Size: 32 bit	**Default:** 10004321h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	EIP	**EHCI Pin:** Indicates which pin the EHCI controller drives as its interrupt. **Bits Pin Bits Pin** 0h No Interrupt 1h INTA# 2h INTB# 3h INTC# 4h INTD# 5h–Fh Reserved		1h	RW
27 : 16	Reserved	Reserved		0h	
15 : 12	Reserved	**Reserved**		4h	
11 : 08	Reserved	**Reserved**		3h	
07 : 04	RSVD	**Reserved.**		2h	RW
03 : 00	U0P	**UHCI 0 Pin:** See the EIP description. Applies to UCHI controller 0 (ports 0 and 1).		1h	RW

17.1.5.3 Offset 3140h: D31IR - Device 31 Interrupt Route Register

Table 17-22. Offset 3140h: D31IR - Device 31 Interrupt Route Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 3140h **Offset End:** 3141h	
Size: 16 bit	**Default:** 3210h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	Reserved	Reserved		0h	
14 : 12	IDR	**Interrupt D Pin Route:** Indicates which physical pin on the IICH is connected to the INTD# pin reported for device 31 functions. **Bits Pin** 0h PIRQA# 1h PIRQB# 2h PIRQC# 3h PIRQD#		3h	RW
11	Reserved	Reserved		0h	
10 : 08	ICR	**Interrupt C Pin Route:** See the IDR description. This field applies to INTC#.		2h	RW
07	Reserved	Reserved		0h	

Table 17-22. Offset 3140h: D31IR - Device 31 Interrupt Route Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3140h **Offset End:** 3141h	
Size: 16 bit	**Default:** 3210h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06 : 04	IBR	**Interrupt B Pin Route:** See the IDR description. This field applies to INTB#.		1h	RW
03	Reserved	Reserved		0h	
02 : 00	IAR	**Interrupt A Pin Route:** See the IDR description. This field applies to INTA#.		0h	RW

17.1.5.4 Offset 3144h: D29IR - Device 29 Interrupt Route Register

Table 17-23. Offset 3144h: D29IR - Device 29 Interrupt Route Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3144h **Offset End:** 3145h	
Size: 16 bit	**Default:** 3210h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	Reserved	Reserved		0h	
14 : 12	IDR	**Interrupt D Pin Route:** Indicates which physical pin on the IICH is connected to the INTD# pin reported for device 29 functions. **Bits Pin** 0h PIRQA# 1h PIRQB# 2h PIRQC# 3h PIRQD#		3h	RW
11	Reserved	Reserved		0h	
10 : 08	ICR	**Interrupt C Pin Route:** See the IDR description. This field applies to INTC#.		2h	RW
07	Reserved	Reserved		0h	
06 : 04	IBR	**Interrupt B Pin Route:** See the IDR description. This field applies to INTB#.		1h	RW
03	Reserved	Reserved		0h	
02 : 00	IAR	**Interrupt A Pin Route:** See the IDR description. This field applies to INTA#.		0h	RW

17.1.5.5 Offset 31FFh: OIC - Other Interrupt Control Register

Table 17-24. Offset 31FFh: OIC - Other Interrupt Control Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 31FFh **Offset End:** 31FFh	
Size: 8 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		0h	
01	CEN	**Coprocessor Error Enable:** 0 = FERR# does not generate IRQ13 nor IGNNE#. 1 = If FERR# is low, the IICH generates IRQ13 internally and holds it until an I/O port F0h write. It also drives IGNNE# active.		0h	RW
00	AEN	**APIC Enable:** 0 = The internal IOxAPIC is disabled. 1 = Enables the internal IOxAPIC and its address decode.		0h	RW

17.1.6 General Configuration Registers

17.1.6.1 Offset 3400h: RC - RTC Configuration Register

Table 17-25. Offset 3400h: RC - RTC Configuration Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3400h **Offset End:** 3403h	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 05	Reserved	Reserved		0h	RO
04	UL	**Upper 128 Byte Lock:** 0 = Bytes 38h-3Fh in the upper 128-byte bank of RTC RAM are not locked and can be accessed. Writes are not dropped and reads return any guaranteed data. 1 = Bytes 38h-3Fh in the upper 128-byte bank of RTC RAM are locked and cannot be accessed. Writes are dropped and reads do not return any guaranteed data. Bit reset on system reset.		0h	RWO

Table 17-25. Offset 3400h: RC - RTC Configuration Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3400h **Offset End:** 3403h	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	LL	**Lower 128 Byte Lock:** 0 = Bytes 38h-3Fh in the lower 128-byte bank of RTC RAM are not locked and can be accessed. Writes are not dropped and reads return any guaranteed data. 1 = Bytes 38h-3Fh in the lower 128-byte bank of RTC RAM are locked and cannot be accessed. Writes are dropped and reads do not return any guaranteed data. Bit reset on system reset.		0h	RWO
02	UE	**Upper 128 Byte Enable:** 0 = The upper 128-byte bank of RTC RAM can not be accessed. 1 = The upper 128-byte bank of RTC RAM can be accessed.		0h	RW
01 : 00	Reserved	Reserved		0h	

17.1.6.2 Offset 3404h: HPTC - High Performance Precision Timer Configuration Register

This register specifies the base address in memory space at which the High Performance Precision Timer registers from Section 32.2.1, “Register Descriptions” materialize.

Table 17-26. Offset 3404h: HPTC - High Performance Precision Timer Configuration Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3404h **Offset End:** 3407h	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 08	Reserved	Reserved		0h	

Table 17-26. Offset 3404h: HPTC - High Performance Precision Timer Configuration Register

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3404h **Offset End:** 3407h	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	AE	**Address Enable:** 0 = The IICH does not decode the High Performance Timer memory address range selected by bits 01:00. 1 = The IICH decodes the High Performance Timer memory address range selected by bits 01:00.		0h	RW
06 : 02	Reserved	Reserved		0h	
01 : 00	AS	**Address Select:** This 2-bit field selects one of four possible memory address ranges for the High Performance Timer functionality. The encodings are: **Bits** **Memory Address Range** 00 FED0_0000h - FED0_03FFh 01 FED0_1000h - FED0_13FFh 10 FED0_2000h - FED0_23FFh 11 FED0_3000h - FED0_33FFh		0h	RW

17.1.6.3 Offset 3410h: GCS: General Control and Status Register

Table 17-27. Offset 3410h: GCS - General Control and Status Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3410h **Offset End:** 3413h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		00h	
23 : 16	BDS	**BIST_Delay_Sel:** This field determines the amount of time, measured in 125 MHz clocks, waits to deassert the INIT# signal after sending the CPU_RESET_DONE_ACK message. ***Notes:*** 1 This field only has meaning if the BIST_EN bit (Bit 2 in the BUC register) is also set. 2 A value of 00h or 01h in this field is not permitted. 3 A 1 clock variation permitted in the actual-time the INIT# signal goes inactive. 4 This field is in the core well. 5 This field is not reset by a CF9 reset with value 06h. Implementation choice: This register does not need to be reset by any reset.		XXh	RW
15 12	Reserved	Reserved		0h	

Table 17-27. Offset 3410h: GCS - General Control and Status Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3410h **Offset End:** 3413h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 : 10	BBS	**Boot BIOS Straps:** This field determines the destination of accesses to the BIOS memory range. 00 = SPI 01 = Reserved 10 = Reserved. 11 = LPC • When SPI or LPC is selected, the range that is decoded is further qualified by other configuration bits described in the respective sections. The value in this field can be overwritten by software as long as the BIOS Interface Lock-Down (bit 0) is not set.		Strap	RW Special
09	SERM	**Server Error Reporting Mode:** 0 =The IICH is the final target of all errors. The IMCH sends a DO_SERR messages to the IICH for the purpose of generating NMI. 1 = The IMCH is the final target of all errors from PCI Express and NSI. In this mode, if the IICH detects a fatal, non-fatal, or correctable error on NSI, it sends one of ERR_FATAL, ERR_NONFATAL, or ERR_CORR to IMCH.		0h	RW
08 : 07	Reserved	Reserved		0h	
06	FME	**FERR# MUX Enable:** This bit enables FERR# to be a CPU break event indication. 0 = does not examine FERR# during a C2, or C4 state as a break event. 1 = IICH examines FERR# during a C2, or C4 state as a break event.		0h	RW
05	NR	**No Reboot:** This bit is set when the “No Reboot” strap is sampled high on PWROK. This bit may be set or cleared by software if the strap is sampled low but may not override the strap when it indicates “No Reboot”. 0 = The TCO timer does not count down and generate the SMI# on the first timeout, but reboots on the second timeout. 1 = The TCO timer counts down and generates the SMI# on the first timeout, but does not reboot on the second timeout.		Strap	RW
04	AME	**Alternate Access Mode Enable:** 0 = Read-only registers cannot be written, and write-only registers cannot be read. See Section 27.6 for details. 1 = Read-only registers can be written, and write-only registers can be read. See Section 27.6 for details.		0h	RW
03	Reserved	Reserved		0b	RO

Table 17-27. Offset 3410h: GCS - General Control and Status Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 3410h **Offset End:** 3413h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	RPR	**Reserved Page Route:** Determines where to send the reserved page registers. These addresses are sent to PCI or LPC for the purpose of generating POST codes. The I/O addresses modified by this field are: 80h, 84h, 85h, 86h, 88h, 8Ch, 8Dh, and 8Eh. 0 = Writes are forwarded to LPC, shadowed within the IICH, and reads are returned from the internal shadow. 1 = Writes are forwarded to PCI, shadowed within the IICH, and reads are returned from the internal shadow. ***Note:*** If some writes are done to LPC/PCI to these I/O ranges, and then this bit is flipped, such that writes now go to the other interface, the reads do not return what was last written. Shadowing is performed on each interface. The aliases for these registers, at 90h, 94h, 95h, 96h, 98h, 9Ch, 9Dh, and 9Eh, are always decoded to LPC.		0h	RW
01	Reserved	Reserved		0h	
00	TSLD	**Top Swap Lock-Down:** 0 = This bit can only be written from 0 to 1 once. BUC.TS can be changed. 1 = Prevents BUC.TS from being changed.		0h	RWO

17.1.6.4 Offset 3414h: BUC - Backed Up Control Register

All bits in this register are in the RTC well and only cleared by RTEST.

Table 17-28. Offset 3414h: BUC - Backed Up Control Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 3414h **Offset End:** 3417h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		0h	

Table 17-28. Offset 3414h: BUC - Backed Up Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3414h **Offset End:** 3417h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	CBE	**CPU BIST Enable:** 0 = The INIT# signal is not driven active when CPURST# is active. 1= The INIT# signal is driven active when CPURST# is active. INIT# goes inactive with the same timings as the other CPU Interface signals (hold time after CPURST# inactive). This bit is in the resume well and is reset by RSMRST#, but not PCIRST# nor CF9h writes.		0h	RW
01	Reserved	Reserved		0h	
00	TS	**Top Swap:** 0 = IICH does not invert A16. 1 = IICH inverts A16 for cycles going to the BIOS space (but not the feature space) in the FWH. If the IICH is strapped for Top-Swap (GNT[6]# is low at rising edge of PWROK), then this bit **cannot** be cleared by software. The strap jumper must be removed and the system rebooted.		Strap	RW

17.1.6.5 Offset 3418h: FD - Function Disable Register

When disabling USB1 host controllers, the USB 2.0 EHCI Structural Parameters Registers must be updated with coherent information in “Number of Companion Controllers” and “N_Ports” fields.

When disabling a function, only the configuration space is disabled. Software must ensure that all functionality within a controller that is not desired (such as memory spaces, I/O spaces, and DMA engines) is disabled prior to disabling the function.

Table 17-29. Offset 3418h: FD - Function Disable Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3418h **Offset End:** 341Bh	
Size: 32 bit	**Default:** 00000080h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		0h	RO
19	Reserved	**Reserved**		0h	RO
18	Reserved	**Reserved**		0h	RO
17	Reserved	**Reserve**d		0h	RO
16	Reserved	**Reserv**ed		0h	RO
15	U2D	**USB 2.0 Disable:** 0 = The USB 2.0 host controller is enabled. 1 = The USB 2.0 host controller is disabled.		0h	RW

Table 17-29. Offset 3418h: FD - Function Disable Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3418h **Offset End:** 341Bh	
Size: 32 bit	**Default:** 00000080h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14	LBD	**LPC Bridge Disable:** 0 = The LPC bridge is enabled. 1 = The LPC bridge is disabled. Unlike the other disables in this register, the following additional spaces no longer are decoded by the LPC bridge: • Memory cycles below 16 MBytes (1000000h) • I/O cycles below 64 Kbytes (10000h) • The Internal I/OxAPIC at FEC0_0000 to FECF_FFFF • Memory cycles in the LPC BIOS range below 4 GByte are still decoded when this bit is set, but the aliases at the top of 1 MByte (the E and F segment) are no longer decoded.		0h	RW
13 : 12	Reserved	Reserved		0h	RO
11	Reserved	**Reserved**		0h	RO
10	Reserved	**Reserved**		0h	RO
09	Reserved	**Reserved**		0h	RO
08	U1D	**USB1 #1 Disable:** 0 = When reset, the first USB 1.1 controller (ports 0 and 1) is enabled. 1 = When set, the first USB 1.1 controller (ports 0 and 1) is disabled.		0h	RW
07	Reserved	Reserved		1h	RO
06	Reserved	**Reserved**		0h	RO
05	Reserved	**Reserved**		0h	RO
04	Reserved	Reserved		0h	RO
03	SD	**SM Bus Disable:** 0 = The SM Bus controller is enabled. 1 = The SM Bus controller is disabled.		0h	RW
02	SAD	**Serial ATA Disable:** 0 = The serial ATA controller is enabled. 1 = The serial ATA controller is disabled		0h	RW
01	Reserved	Reserved		0h	RW
00	Reserved	**Reserved**			

17.1.6.6 Offset 341Ch: PRC - Power Reduction Control Register Clock Gating

Table 17-30. Offset 341Ch: PRC - Power Reduction Control Register Clock Gating

Description:					
View: PCI	**BAR:** RCBA	**Bus:Device:Function:** 0:31:0		**Offset Start:** 341Ch **Offset End:** 341Fh	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 01	Reserved	Reserved		0h	
00	Reserved	**Reserved**			

§ §

18.0 System Management

18.1 Overview

The CMI provides various functions to make a system easier to manage and lower the Total Cost of Ownership (TCO) of the system. Features and functions can be augmented via external A/D converters and GPIO, as well as an external microcontroller.

The following features and functions are supported by CMI:

- First timer to generate SMI# after programmable time.
  - — First timeout causes SMI#; allows for SMM-based recovery from operating system lockup.
  - — Operating system-based software agent accesses CMI to periodically reload timer.
- Ability for SMM handler to generate “TCO” interrupt to operating system.
  - — Allows for operating system-based code augmentation.
- Ability for operating system to generate SMI#.
  - — Call-back from operating system to TCO code in SMM handler.
- Second hard coded timeout to generate reboot.
  - — Used only after first timeout occurs.
  - — Second timeout allows for automatic system reset and reboot if hardware error detected. Various system states are preserved via this special reset to allow for possible error detection and correction.
  - — Reset associated with reboot may attempt to preserve some registers for diagnostic purposes.
  - — SMI# handler must reload the main timer within 2.4 s to prevent the second timer from causing a reboot (timeout during SMI is assumed as broken CPU or stuck hardware).
  - — Option to prevent reset if second timeout occurs.
- Processor present detection.
  - — Detects if processor fails to fetch the first instruction after reset.
  - — If CPU failure detected, option to pulse a GPIO or send SMBus message. The SMBus message can be used to indicate to an External LAN controller to send a distress message. The GPIO can control an LED with optional blink.
- Ability to handle various errors (such as ECC errors) indicated by the IMCH.
  - — Can generate SMI# or TCO interrupt.
- Intruder detect input.
  - — Can generate TCO interrupt or SMI# when the system cover is removed.
- Ability for TCO messages to coexist with standard SMBus devices.
- Detection of bad FWH programming.

— Detects if data on first read is FFh (indicates unprogrammed Firmware Hub).

18.2 TCO I/O-Mapped Configuration Register Details

The TCO logic is accessed via registers mapped to the PCI configuration space (Device 31, Function 0) and the system I/O space. For TCO PCI Configuration registers, see Section 19.0, “LPC Interface: Bus 0, Device 31, Function 0”.

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183).

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values and are read only. Writes to reserved locations may cause system failure and unpredictable results.

Note: Reserved bits are Read Only.

Table 18-1. Bus 0, Device 31, Function 0: Summary of TCO Configuration Registers Mapped Through TCOBASE I/O BAR“

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	“Offset 00h: TRLD - TCO Timer Reload and Current Value Register” on page 715	0000h
02h	02h	“Offset 02h: TDI - TCO Data In Register” on page 715	00h
03h	03h	“Offset 03h: TDO - TCO Data Out Register” on page 716	00h
04h	04h	“Offset 04h: TSTS1 - TCO 1 Status Register” on page 716	0000h
06h	07h	“Offset 06h: TSTS2 - TCO 2 STS Register” on page 718	0000h
08h	09h	“Offset 08h: TCTL1 - TCO 1 Control Register” on page 720	0000h
0Ah	0Bh	“Offset 0Ah: TCTL2 - TCO 2 Control Register” on page 721	0008h
0Ch at 01h	0Ch at 01h	“Offset 0Ch: TMSG[1-2] - TCO MESSAGE Register” on page 721	00h
0Eh	0Eh	“Offset 0Eh: TWDS - TCO Watchdog Status Register” on page 722	00h
10h	10h	“Offset 10h: LE - Legacy Elimination Register” on page 722	03h
12h	13h	“Offset 12h: TTMR - TCO Timer Initial Value Register” on page 723	0004h

18.2.1 TCO PCI Configuration Registers

See Chapter 19.0, "LPC Interface: Bus 0, Device 31, Function 0."

Allows setting of the Base Address for the I/O space and routing of the TCO interrupt.

18.2.2 Bus 0, Device 31, Function 0: TCO Configuration Register (I/O-Mapped via ABASE BAR) Summary Table

The TCO I/O registers reside in a 32-byte range that starts 96 bytes above the power management (ACPI) I/O space (Section 19.2.2.1, "Offset 40h: ABASE: ACPI Base Address Register"). Thus **TCOBASE (IO) =ABASE (IO) + 60h** in the PCI configuration space. Table 18-2 shows the mapping of the registers within that 32-byte range. Each register is described in the following sections.

18.2.2.1 Offset 00h: TRLD - TCO Timer Reload and Current Value Register

Table 18-2. Offset 00h: TRLD - TCO Timer Reload and Current Value Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	**Reserved**.		00h	RO
09 : 00	TRLD	**TCO Timer Value:** Reading this register returns the current count of the TCO timer. Writing any value to this register reloads the timer to prevent the timeout.		000h	RW

18.2.2.2 Offset 02h: TDI - TCO Data In Register

Table 18-3. Offset 02h: TDI - TCO Data In Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	TDI	This data register field is used for passing commands from the operating system to the SMI handler. Writes to this register cause an SMI and set the OS_TCO_SMI bit in the TCO_STS register (D31, F0, 04h).		00h	RW

18.2.2.3 Offset 03h: TDO - TCO Data Out Register

Table 18-4. Offset 03h: TDO - TCO Data Out Register

Description:					
View: PCI	BAR: TCOBASE (IO)		Bus:Device:Function: 0:31:0	Offset Start: 03h Offset End: 03h	
Size: 8 bit	Default: 00h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	TDO	This data register field is used for passing commands from the SMI handler to the operating system. Writes to this register sets the TCO_INT_STS bit in the TCO_STS register. It also causes an interrupt, as selected by the TCO_IRQ_SEL bits.		00h	RW

18.2.2.4 Offset 04h: TSTS1 - TCO 1 Status Register

Unless otherwise indicated, these bits are “sticky” and are cleared by writing a one to the corresponding bit position.

Table 18-5. Offset 04h: TSTS1 - TCO 1 Status Register (Sheet 1 of 3)

Description:					
View: PCI	BAR: TCOBASE (IO)		Bus:Device:Function: 0:31:0	Offset Start: 04h Offset End: 04h	
Size: 16 bit	Default: 0000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 13	Reserved	Reserved		000h	RO
12	MCHSERR_STS	0 = Software clears this bit by writing a 1 to it. 1 = The IMCH sent a special cycle message NSI indicating that it wants to cause an SERR#. The software must read the IMCH to determine the reason for the SERR#.		0h	RWC
11	Reserved	Reserved.		0h	RO
10	MCHSMI_STS	0 = Software clears this bit by writing a 1 to it. 1 = IMCH sends a special cycle message indicating that it wants to cause an SMI. The software must read the IMCH to determine the reason for the SMI.		0h	RWC
09	MCHSCI_STS	0 = Software clears this bit by writing a 1 to it. 1 = IMCH sends a special cycle message indicating that it wants to cause an SCI. The software must read the IMCH to determine the reason for the SCI.		0h	RWC

Table 18-5. Offset 04h: TSTS1 - TCO 1 Status Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 04h **Offset End:** 04h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	BIOSWR_STS	0 = Software clears this bit by writing a 1 to it. 1 = The CMI sets this bit and generates and SMI# to indicate an illegal attempt to write to the BIOS. This occurs when either: a) The BIOSWP bit is changed from 0 to 1 and the BLD bit is also set, or b) any write is attempted to the BIOS and the BIOSWP bit is also set. ***Note:*** On write cycles attempted to the 4 Mbyte lower alias to the BIOS space, the BIOSWR_STS is not set.		0h	RWC
07	NEWCENTURY_STS	This bit is in the RTC well. 0 = Cleared by writing a 1 to the bit position or by RTEST# going active. 1 = This bit is set when the Year byte (index offset 09h) rolls over from 1999 to 2000. If the bit is already 1, it remains 1. When this bit is set, an SMI# is generated. However, this is not a wake event (i.e., if the system is in a sleeping state when the NEWCENTURY_STS bit is set, the system does not wake up). ***Note:*** The NEWCENTURY_STS is not valid when the RTC battery is first installed (or if the RTC battery does not provide sufficient power when the system is unplugged). Software can determine that the RTC well was not maintained by checking the RTC_PWR_STS bit (D31:F0:A4, bit 2) or by other means (such as doing a checksum on the RTC RAM array). If the RTC power is determined to not have been maintained, the BIOS must set the time to a legal value and then clear the NEWCENTURY_STS bit. The NEWCENTURY_STS bit may take up to 3 RTC clocks for the bit to be cleared when a 1 is written to the bit to clear it. After writing a 1 to the NEWCENTURY_STS bit, software must not exit the SMI handler until verifying that the bit has actually been cleared. This ensures that the SMI is not reentered.		0h	RWC
06 : 04	Reserved	Reserved		0h	RO
03	TIMEOUT	0 = Software clears this bit by writing a 1 to it. 1 = Set to indicate that the SMI was caused by the TCO timer reaching 0. ***Note:*** The SMI handler must clear this bit to prevent an immediate reentry to the SMI handler.		0h	RWC

Table 18-5. Offset 04h: TSTS1 - TCO 1 Status Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 04h **Offset End:** 04h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	TCO_INT_STS	0 = Software clears this bit by writing a 1 to it. 1 = SMI handler caused the interrupt by writing to the TCO_DAT_OUT register (TCOBASE + 03h).		0h	RWC
01	OS_TCO_SMI	0 = Software clears this bit by writing a 1 to it. 1 = Software caused an SMI# by writing to the TCO_DAT_IN register (TCOBASE + 02h).		0h	RWC
00	NMI2SMI_STS	0 = Cleared by clearing the associated NMI# status bit. 1 = Set when an SMI# occurs because an event occurred that would otherwise have caused an NMI#. ***Note:*** The NMI2SMI_STS bit must not be “sticky bit”. It must be a simple OR gate to indicate that one of the NMI sources has caused the SMI. Each of the NMI sources already has its own sticky bit feeding the OR gate. ***Note:*** Writes to this bit have no effect.		0h	RWC

18.2.2.5 Offset 06h: TSTS2 - TCO 2 STS Register

Table 18-6. Offset 06h: TSTS2 - TCO 2 STS Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 05	Reserved	Reserved		000h	RO
04	SMLINK_ SLAVE_ SMI_STS	Allows the software to go directly into predetermined sleep state. This avoids race conditions. Software clears this bit by writing a 1 to it. 0 = The bit is reset by RSMRST#, but not due to the PCI Reset associated with exit from S3–S5 states. 1 = The CMI sets this bit to 1 when it receives the SMI message (encoding 08h in the command type) on the SMLink's Slave Interface. This bit is in the resume well. It is reset by RSMRST#–		0h	RWC
03	BAD_BIOS	This bit is not intended to be read by the BIOS or software. It is only used for sending the TCO messages to an External LAN Controller. 0 = The first BIOS read is not FFh. This is detected when the initial read returns FFh from the FWH. Reads to this bit always return 0 and writes have no effect. 1 = FFh is detected on the first BIOS read (i.e., the BIOS is bad).		0h	RO

Table 18-6. Offset 06h: TSTS2 - TCO 2 STS Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	DOACPU_ STS	0 = Cleared based on RSMRST# or by software writing a 1 to this bit. Software must first clear the SECOND_TO_STS bit before writing a 1 to clear the BOOT_STS bit. 1 = Set to 1 when the SECOND_TO_STS bit goes from 0 to 1 and the processor has not fetched the first instruction. If rebooting due to a SECOND_TO_STS bit set (= 1) and the DOACPU_STS bit is: 0 = The BIOS can conclude that the system rebooted due to some lockup (such as on NSI), but not due to a processor booting issue. 1 = Reboots using the 'safe' multiplier (1111). This allows the system to recover from a processor frequency multiplier that is too high, and allows the BIOS to check the DOACPU_STS bit at boot. If the bit is set and the frequency multiplier is 1111, then the BIOS knows that the processor has been programmed to an illegal multiplier. ***Note:*** Software must clear the SECOND_TO_STS bit first, then the DOACPU_STS bit (use two separate I/O write operations).		0h	RWC
01	SECOND_ TO_STS	0 = Software clears this bit by writing a 1 to it or by a RSMRST#. 1 = Sets this bit to 1 to indicate that the TIMEOUT bit had been (or is currently) set and a second timeout occurred before the TCO_RLD register was written. If this bit is set and the NO_REBOOT configuration bit is 0, then reboots the system after the second timeout. The reboot is done by asserting PLTRST#.		0h	RWC
00	INTRD_DET	**Intruder Detect.** This bit resides in the RTC well. 0 = Software clears this bit by writing a 1 to this bit or by RTEST#. 1 = Set to indicate that an intrusion was detected. This bit is latched. The INTRUDER# signal must be asserted for a minimum of 1 ms to guarantee that the INTRD_DET bit is set. This bit has a recovery time. After writing a 1 to this bit position (to clear it), the bit may be read back as a 1 for up 65 µs before it is read as a 0. Software must be aware of this recovery time when reading this bit after clearing it. If the INTRUDER# signal is active when the software attempts to clear the INTRD_DET bit, the bit remains one, and the SMI# is generated again immediately. The SMI handler can clear the INTRD_SEL bits (TCOBASE + 0Ah, bits 2:1) to avoid further SMIs. However, if the INTRUDER# signals goes inactive and then active again, there is not further SMIs (because the INTRD_SEL bits would select that no SMI# be generated). If the INTRUDER# signal goes inactive some point after the INTRD_DET bit is written as a 1, then the INTRD_DET signal goes to a 0 when INTRUDER# input signal goes inactive. This is slightly different than a classic sticky bit, since most sticky bits would remain active indefinitely when the signal goes active and would immediately go inactive when a 1 is written to the bit.		0h	RWC

18.2.2.6 Offset 08h: TCTL1 - TCO 1 Control Register

Table 18-7. Offset 08h: TCTL1 - TCO 1 Control Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 08h **Offset End:** 09h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 13	Reserved	Reserved		000h	RO
12	TCO_LOCK	0 = A core well reset is required to change this bit from 1 to 0. This bit defaults to 0. 1 = This bit prevents writes from changing the TCO_EN bit (in offset 30h of Power Management I/O space). Once this bit is set to 1, it can not be cleared by software writing a 0 to this bit location.		0h	RW
11	TCO_TMR_HALT	0 = The TCO timer is enabled to count. 1 = The TCO Timer halts. It does not count, and thus cannot reach a value that causes an SMI# or set the SECOND_TO_STS bit. When set, this bit prevents rebooting.		0h	RW
10	SEND_NOW	0 = Clears this bit when it has completed sending the message. Warning: Software must not set this bit to 1 again until the CMI has set it back to 0. 1 = Set the SEND_NOW bit and causes the LAN controller to reset, which can have unpredictable side-effects. Unless software protects against these side-effects, software must not set or otherwise use the SEND_NOW bit.		0h	RW
09	NMI2SMI_EN	0 = Normal NMI functionality 1 = Setting this bit 1 forces all NMIs to instead cause an SMI#, and is reported in the TCO1_STS register. NMI2SMI_EN bit is set AND the NMI_EN bit is set, the NMI# is routed to cause an SMI#. No NMI is caused. However, if the GBL_SMI_EN bit is not set, then no SMI# is generated, either. If NMI2SMI_EN is set but the NMI_EN bit is not set, then no NMI or SMI# is generated. The following table shows the possible combinations: NMI_EN / GBL_SMI_EN / Description 0b / 0b / No SMI# at all because GBL_SMI_EN = 0 0b / 1b / SMI# is caused due to NMI events 1b / 0b / No SMI# at all because GBL_SMI_EN = 0 1b / 1b / No SMI# due to NMI because NMI_EN = 1		0h	RW
08	NMI_NOW	0 = Software clears this bit by writing a 1 to it. The NMI handler is expected to clear this bit. Another NMI is not generated until the bit is cleared. 1 = Writing a 1 to this bit causes an NMI. This allows the BIOS or SMI handler to force an entry to the NMI handler.		0h	RWC
07 : 00	Reserved	Reserved		00h	RO

18.2.2.7 Offset 0Ah: TCTL2 - TCO 2 Control Register

Table 18-8. Offset 0Ah: TCTL2 - TCO 2 Control Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0Ah **Offset End:** 0Bh	
Size: 16 bit	**Default:** 0008h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 06	Reserved	Reserved		000h	RO
05 : 04	OS_POLICY	Operating system-based software writes to these bits to select the policy that the BIOS uses after the platform resets due the WDT. The following convention is recommended for the BIOS and operating system: 00 Boot normally 01 Shut down 10 Do not load operating system. Hold in preboot state and use LAN to determine next step 11 Reserved ***Note:*** These are scratchpad bits. They must not be reset when the TCO logic resets the platform due to Watchdog Timer.		00h	RW
03	GPIO11_ ALERT_ DISABLE	At reset (via RSMRST# asserted) this bit is set and GPI[11] alerts are disabled. 0 = Enable 1 = Disable GPI[11]/SMBALERT# as an alert source for the SMBus slave		1	RW
02 : 01	INTRD_SEL	This field selects the action to take if the INTRUDER# signal goes active. 00 No interrupt or SMI# 01 Interrupt (as selected by TCO_INT_SEL). 10 SMI# 11 Reserved		00h	RW
00	Reserved	Reserved		0h	RO

18.2.2.8 Offset 0Ch: TMSG[1-2] - TCO MESSAGE Register

Note: The following table represents two registers.

Table 18-9. Offset 0Ch: TMSG[1-2] - TCO MESSAGE Register

Description: Offset: TMSG[1] = 0Ch-0Ch; TMSG[2] = 0Dh-0Dh					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0Ch at 01h **Offset End:** 0Ch at 01h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	TMSG	The value written into this register is sent out in the MESSAGE field of the SMBus Event messages. BIOS can write to this register to indicate its boot progress which can be monitored externally.		00h	RW

18.2.2.9 Offset 0Eh: TWDS - TCO Watchdog Status Register

Table 18-10. Offset 0Eh: TWDS - TCO Watchdog Status Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	TWDS	The value written to this register is passed via SMBus to an External LAN controller. It can be used by the BIOS or system management software to indicate more details on the boot progress. The register is reset to 00h based on a RSMRST# (but not PCI reset).		00h	RW

18.2.2.10 Offset 10h: LE - Legacy Elimination Register

Table 18-11. Offset 10h: LE - Legacy Elimination Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 10h **Offset End:** 10h	
Size: 8 bit	**Default:** 03h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		00h	RO
01	IRQ12_CAUSE	0 = When software sets the bit to 0, IRQ12 is low (not asserted). 1 = When software sets the bit to 1, IRQ12 is high (asserted).		1	RW
00	IRQ1_CAUSE	0 = When software sets the bit to 0, IRQ1 is low (not asserted). 1 = When software sets the bit to 1, IRQ1 is high (asserted).		1	RW

18.2.2.11 Offset 12h: TTMR - TCO Timer Initial Value Register

Table 18-12. Offset 12h: TTMR - TCO Timer Initial Value Register

Description:					
View: PCI	**BAR:** TCOBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 12h **Offset End:** 13h	
Size: 16 bit	**Default:** 0004h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 10	Reserved	Reserved		00h	RO
09 : 00	TTMR	Value that is loaded into the timer each time the TCO_RLD register is written. Values of 0000h or 0001h are ignored and must not be attempted. The timer is clocked at approximately 0.6 s, and thus allows timeouts ranging from 1.2 s to 613.8 s. ***Note:*** The timer has an error of +/- 1 tick (0.6 s). The TCO Timer only counts down in the S0 state.		004h	RW

18.3 TCO Signal Usage

18.3.1 INTRUDER# Signal

This signal can be used to detect the chassis being opened. The activation of this signal can be used to cause an SMI#, and is reported via the event mechanism. If SMI# is desired, the signals level can be read, so this can be used as a type of general purpose input.

18.3.2 Pin Straps

Some of the TCO functions are decided at power up (rising edge of PWROK). See the pinlist for specific assignments of pin straps.

18.3.3 SMLINK Signals

The CMI supports TCO compatible mode connectivity. TheIICHsupports External LAN controllers. An external LAN Controller can be used to receive or retrieve TCO message or information on Host SMBus if needed. In Legacy TCO mode messages are driven via SMLink.

For the CMI, messages on this link use SMBus protocol at the rates described in Chapter 24.0, “SMBus Controller Functional Description: Bus 0, Device 31, Function 3,” for TCO compatible mode.

18.4 TCO Theory of Operation

18.4.1 Overview

The system management functions are designed to allow the system to diagnose failing subsystems. The intent of this logic is that the system management functionality be provided without the aid of an external micro controller.

The CMI’s System Management logic allows for diagnostic and recovery software to be distributed between an SMI handler and operating system-based code.

18.4.2 Detecting a DOA CPU or System

When the processor is reset, it is expected to fetch its first instruction. If the processor fails to fetch the first instruction after reset, the TCO timer times out twice and CMI asserts PLTRST#.

If TCO Reboots are not enabled, then:

a. The SMLink sends out the first eight bits of the message. After the eighth bit, the logic stalls because there is no integrated LAN controller to send the ACK. The logic then aborts the transfer. External logic monitors the toggling and use that to drive an LED.

b. If an External LAN controller is connected send the appropriate message to the External LAN controller.

If TCO Reboots are enabled, then the CMI attempts to reboot the system.

1. If the NO-REBOOT bit (NR field in Table 17-27, “Offset 3410h: GCS - General Control and Status Register” on page 706) is set (no reboots are intended)

and

SECOND_TO_STS bit (TCO I/O Offset 06h, bit 1) is set

and

DOACPU_STS bit (TCO I/O Offset 06h, bit 2) is set,

then the CMI indicates this in the TCO message by setting the CPU Missing bit in the message.

2. If the NO-REBOOT bit (NR field in Table 17-27, “Offset 3410h: GCS - General Control and Status Register” on page 706) is not set (reboots intended)

and

SECOND_TO_STS bit TCO I/O Offset 06h, bit 1) is set,

then the CMI attempts to reboot. After the reboot, the SECOND_TO_STS bit is still set. If the CPU fails to fetch the first instruction, the DOA_CPU_STS bit is set, and when the TCO timer times out (actually for the third time, the first two caused the SECOND_TO_STS bit to be set), then the CMI sets the CPU MISSING EVENT bit for the TCO message.

18.4.3 Handling an Operating System Lockup

Under some conditions, the operating system may lock up. To handle this, the TCO Timer is used with the following algorithm:

1. BIOS programs the TCO Timer, via the TCO_TMR register with an initial value.
2. An operating system-based software agent periodically writes to the TCO_RLD register to reload the timer and keep it from generating the SMI#. The software agent can read the TCO_RLD register to see if it is close to timing out, and possibly determine if the time-out should be increased. The operating system can also modify the values in the TCO_TMR register.
3. If the timer reaches 0, an SMI# can be generated. This should only occur if the operating system was not able to reload the timer. It is assumed that the operating system is not able to reload the timer if it has locked up.
4. Upon generating the SMI#, the TCO Timer automatically reloads with the default value of 04h and start counting down.

5. The SMI handler can then:
   a. Read the TIMEOUT bit in the TCO_STS register to check that the SMI# was caused by the TCO timer. The SMI handler must also clear the TIMEOUT bit.
   b. Write to the TCO_RLD register to reload the timer to make sure the TCO timer does not reach 0 again.
   c. Attempt to recover. May need to periodically reload the TCO timer.

The exact recovery algorithm is system-specific.

Note: If the SMI handler was not able to clear the TIMEOUT bit and write to the TCO_RLD register, the timer reaches zero a second time approximately 2.4 s later. At that point, the hardware is assumed to be locked up, and the timer reads zero a second time, which causes the SECOND_TO_STS bit to be set. At that point the logic resets the platform if the reboots are enabled.

18.4.4 Handling a CPU or Other Hardware Lockup

If after the TIMEOUT SMI is generated, and the TCO timer again reaches 0, and reboots are enabled, the System Management logic resets (and reboot) the system. This is in the case where the CPU or other system hardware is locked up. During every boot, BIOS must read the SECOND_TO_STS bit in the TCO_STS register to see if this is normal boot or a reboot due to the timeout. The BIOS may also check the OS_POLICY bits to see if it should try another boot or shutdown.

18.4.5 Handling an Intruder

The CMI has an input signal, INTRUDER#, that can be attached to a switch that is activated by the system's case being open. This input has a two RTC clock debounce. If INTRUDER# goes active (after the debouncer), this sets the INTRD_DET bit in the TCO_STS register. INTRUDER# can go active in any power state.

The INTRD_SEL bits in the TCO_CNT register can enable the CMI to cause an SMI# or TCO interrupt. The software can also directly read the status of the INTRUDER# signal (high or low) by clearing and then reading the INTRD_DET bit. This allows the signal to be used as a GPI if the intruder function is not required.

Note: The INTRD_DET bit resides in the CMI's RTC well, and is set and cleared synchronously with the RTC clock. Thus, when software attempts to clear INTRD_DET (by writing a one to the bit location) there may be as much as two RTC clocks (about 65 µs) delay before the bit is actually cleared. Also, the INTRUDER# signal must be asserted for a minimum of 1 ms to guarantee that the INTRD_DET bit is set.

Note: If the INTRUDER# signal is still active when software attempts to clear the INTRD_DET bit, the bit remains set and the SMI is generated again immediately. The SMI handler can clear the INTRD_SEL bits to avoid further SMIs. However, if the INTRUDER# signal goes inactive and then active again, there is not further SMIs, since the INTRD_SEL bits would select that no SMI# be generated.

18.4.6 Handling a Potentially Failing Power Supply

It may be possible to detect that a power supply may fail in the near future by monitoring its voltages for fluctuations. To support such an application, external A/Ds with programmable thresholds could be included via SMBus/I^2C. Upon receiving the SMBus/I^2C message, the CMI can generate an SMI or interrupt. The SMI handler (or operating system-based extension) could then attempt to send a message before the power completely fails.

Another option would be to build an A/D into the power supply itself. Another signal, other than PWROK, could report that the power supply might soon fail.

18.4.7 Handling an ECC Error or Other Memory Error

The IMCH provides a message to indicate that it would like to cause an SMI#, SCI, SERR#, or NMI. The software must check the IMCH as to the exact cause of the error.

18.4.8 SMM to Operating System and Operating System to SMM Calls

There may be interaction between an SMI handler and operating system-related code. Two 8-bit data registers are provided.

1. The SMI handler generates an interrupt to the operating system by writing to the TCO_DAT_OUT register. This sets the TCO_INT_STS bit in the TCO_STS register. The interrupt is cleared by writing a one to the TCO_INT_STS bit.
2. The operating system (or driver) can generate an SMI# by writing to the TCO_DAT_IN register. This sets the OS_SMI_STS bit in the TCO_STS register. The SMI# is cleared by writing a one to the OS_SMI_STS bit.

Reads to the TCO_DAT_IN and TCO_DAT_OUT register do not effect the SMI# or INTERRUPT.

Writing a one to the NMI_NOW bit allows for an immediate NMI.

18.4.9 Detecting an Improper FWH Programming

The CMI can detect the case where the FWH is not programmed. This results in the first instruction fetched to have a value of FFh. If this occurs, the CMI sets the BAD_BIOS bit.

18.4.10 IRQ1 and IRQ12 for Legacy Elimination

The new IRQ1 and IRQ12 sources are each logically ANDed with the respective IRQ1 and IRQ12 that come from the SERIRQ logic. This is necessary because the SERIRQ logic reports IRQ1 and IRQ12 to be high (active), since there is no Super I/O to drive them low.

Note: In a system that does have a Super I/O, the new bits must be left at one, since it is ANDed with the Super I/Os IRQ. **Do not attempt to write these bits to 0 in a system that has a keyboard controller (such as in a Super I/O).** It is not validated, and is highly likely to cause errors.

The following algorithm assumes the byte is being sent from the keyboard. The byte being sent from the mouse is equivalent. The setup to the area of interest is left at a high level in this description. The area of interest is then described in more detail.

1. An SMI is received and discovered to be a USB interrupt.
2. The interrupt is discovered to be due to a TD associated with a keyboard device.
3. The data is analyzed and it is determined that the interrupt is due to a new key press.
4. The USB key-code is translated into the equivalent scan code set 2 (SS2) PS/2 scan code.
5. The result is queued on a queue of data to be sent from the keyboard to the system.
6. Other USB interrupts are handled.

7. The keyboard controller emulation code is executed at exit. It determines if the conditions are correct to return a byte to the system (e.g., emulated OBF indicates empty, keyboard interface not disabled, etc.). If not, the emulation exits awaiting the next event.
8. The queue of data to be sent from the keyboard to the system is found to contain a byte to be returned.
9. Given the typical keyboard controller configuration, it is translated from SS2 to SS1.

End of Setup.

10. The byte to be returned is stored in the emulated output buffer.
11. The emulation does the out to the port to enable IRQ 1.
12. The emulation exits.
13. time passes
14. The system code services the interrupt and reads port 60h.
15. The UHCI traps the read and causes an SMI trap.
16. The trap is determined to be caused by the read from port 60h (TBY60R in LEGSUP).
17. The emulation code clears the interrupt register thus turning off IRQ 1 and / or IRQ 12.

Note: The emulation code can validly do this each time there is a read from 60h since that is what the keyboard controller would do as well. (There is no time where both IRQ 1 and IRQ 12 must be asserted simultaneously by either the kbc or the emulated kbc. This would be a violation of the kbc/system protocol.)

18.5 Event Reporting via SMLink/SMBus

The CMI has SMLINK signals to support TCO compatible mode. Event reporting is accomplished via the SMLINK signals.

18.5.1 Overview

18.5.1.1 TCO Compatible Mode

The CMI can function directly with a LAN controller to report message to a network management console without the aid of the system CPU. This is crucial in cases where the CPU is malfunctioning or cannot function due to being in a low-power state.

The basic scheme is to send specific messages via the SMLink Interface to the LAN. Upon receiving the SMLink message, the LAN has a prepared ethernet message that it can send to a network management console. The prepared message is stored in a non-volatile memory connected directly to the LAN.

Messages are sent by the CMI to the LAN either because a specific event has occurred (see Table 18-13), or they are sent periodically. The event messages have exactly the same form.

Table 18-13. Event Transitions that Cause Messages

Event	Assertion	Deassertion	Comments
INTRUDER# pin	yes	no	
PROCHOT# pin	yes	yes	The PROCHOT# pin is isolated when the core power is off, thus preventing this event in S3,S5.
Watchdog Timer Expired	yes	no (NA)	
SEND_NOW bit	yes	NA	Occurs in G0
GPIO[11]/SMBALERT# pin	yes	yes	
BATLOW#	yes	yes	
CPU_PWR_FLR	yes	no	

Note: The GPIO[11]/SMBALERT# pin triggers an event message (when enabled by the GPIO11_ALERT_DISABLE bit) regardless of whether it is configured as a GPI or not.

Whenever an event occurs that causes the CMI to send a new message, it increments the SEQ[03:00] field.

If a triggering event occurs while a message is already being generated and sent, the new event may not appear in the current message. If not, then a second message is generated, the SEQ[03:00] field increments to report the new event.

The following rules/steps apply if the system is in a G0 state and the policy is to reboot the system after a hardware lockup:

1. Upon detecting the lockup the SECOND_TO_STS bit is set. The CMI may send up to one event message to the LAN. The CMI then attempts to reboot the CPU.
2. If the reboot at step 1 is successful then the BIOS must clear the SECOND_TO_STS bit. This prevents any further messages from being sent. The BIOS may then perform addition recovery/boot steps.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the alerts interfere with the LAN device driver from working properly. The alerts reset part of the LAN and would prevent an operating system's device driver from sending or receiving some messages.

3. If the reboot attempt in step 1 is not successful, then the timer timeouts a third time. At this point the system has locked up and was unsuccessful in rebooting. The CMI does not attempt to automatically reboot again. The CMI starts sending a message every period (30-32 seconds). This continues until some external intervention occurs (reset, power failure, etc.).
4. After step 3 (unsuccessful reboot after third timeout), if the user presses a Power Button Override, the system goes to an S5 state. The CMI continues sending the messages every period.
5. After step 4 (power button override after unsuccessful reboot) if the user presses the Power Button again, the system must wake to an S0 state and the CPU must start executing the BIOS.
6. If step 5 (power button press) is successful in waking the system, the CMI continues sending messages every period until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the messages interfere with the LAN device driver from working properly. The alerts reset part of the LAN and prevents an operating system's device driver from sending or receiving some LAN packets.

7. If step 5 (power button press) is unsuccessful in waking the system, the CMI continues sending a message every period. The CMI does not attempt to automatically reboot again. The CMI starts sending a message every period (30-32 seconds). This continues until some external intervention occurs (reset, power failure, etc.).

Note: A system that has locked up and can not be restarted with the power button press is assumed to have broken hardware (bad power supply, short circuit on some bus, etc.), and is beyond the CMI's recovery mechanisms.

8. After step 3 (unsuccessful reboot after third timeout), if a reset is attempted (using a button that pulses PWROK low or via the message on the SMBus Slave Interface), the CMI attempts to reset the system.
9. After step 8 (reset attempt) if the reset is successful, then the BIOS is run. The CMI continues sending a message every period until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the messages interfere with the LAN device driver from working properly. The alerts reset part of the LAN and prevent an operating system's device driver from sending or receiving some LAN packets.

10. After step 8 (reset attempt), if the reset is unsuccessful, then the CMI continues sending a message every period. The CMI does not attempt to reboot the system again without external intervention.

Note: A system that has locked up and can not be restarted with the power button press is assumed to have broken hardware (bad power supply, short circuit on some bus, etc.), and is beyond the CMI's recovery mechanisms.

The following rules/steps apply if the system is in a G0 state and the policy is for the CMI to **not** reboot the system after a hardware lockup:

1. Upon detecting the lockup the SECOND_TO_STS bit is set. The CMI sends a message with the Watchdog (WD) Event status bit set (and any other bits that must also be set). This message is sent as soon as the lockup is detected, and is sent with the next (increment) sequence number.
2. After step 1, the CMI sends a message every period until some external intervention occurs.
3. Rules/steps 4-10 apply if no user intervention (resets, power button presses, SMBus reset messages) occur after a third timeout of the watchdog timer. If the intervention occurs before the third timeout, then jump to step11.
4. After step 3 (third timeout), if the user does a Power Button Override, the system goes to an S5 state. The CMI continues sending messages at this point.
5. After step 4 (power button override), if the user presses the power button again, the system must wake to an S0 state and the CPU must start executing the BIOS.
6. If step 5 (power button press) is successful in waking the system, the CMI continues sending messages until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the alerts interfere with the LAN device driver from working properly. The alerts reset part of the LAN and would prevent an operating system's device driver from sending or receiving some messages.

7. If step 5 (power button press) is unsuccessful in waking the system, the CMI continues sending messages. The CMI does not attempt to reboot the system again until some external intervention occurs (reset, power failure, etc.).

Note: A system that has locked up and can not be restarted with power button press is probably very broken (bad power supply, short circuit on some bus, etc.) and beyond the CMI's recovery mechanisms.

8. After step 3 (third timeout), if a reset is attempted (using a button that pulses PWROK low or via the message on the SMBus Slave Interface), the CMI attempts to reset the system.
9. If step 8 (reset attempt) is successful, then the BIOS is run. The CMI continues sending messages until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the alerts interfere with the LAN device driver from working properly. The alerts reset part of the LAN and would prevent an operating system's device driver from sending or receiving some messages.

10. If step 8 (reset attempt), is unsuccessful, then the CMI continues sending messages. The CMI does not attempt to reboot the system again without external intervention.

Note: A system that has locked up and can not be restarted with the power button press is broken (bad power supply, short circuit on some bus, etc.)

11. This and the following rules/steps apply if the user intervention (power button press, reset, SMBus message, etc.) occur prior to the third timeout of the watchdog timer.
12. After step 1 (second timeout), if the user does a Power Button Override, the system goes to an S5 state. The CMI continues sending messages at this point.
13. After step 12 (power button override), if the user presses the power button again, the system must wake to an S0 state and the CPU must start executing the BIOS.
14. If step 13 (power button press) is successful in waking the system, the CMI continues sending messages until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the alerts interfere with the LAN device driver from working properly. The alerts reset part of the LAN and would prevent an operating system's device driver from sending or receiving some messages.

15. If step 13 (power button press) is unsuccessful in waking the system, the CMI continues sending messages. The CMI does not attempt to reboot the system again until some external intervention occurs (reset, power failure, etc.).

Note: A system that has locked up and can not be restarted with power button press is broken (bad power supply, short circuit on some bus, etc.) and beyond the CMI's recovery mechanisms.

16. After step 1 (second timeout), if a reset is attempted (using a button that pulses PWROK low or via the message on the SMBus Slave Interface), the CMI attempts to reset the system.
17. If step 16 (reset attempt) is successful, then the BIOS is run. The CMI continues sending messages until the BIOS clears the SECOND_TO_STS bit.

Warning: It is important the BIOS clears the SECOND_TO_STS bit, as the alerts interfere with the LAN device driver from working properly. The alerts reset part of the LAN and would prevent an operating system's device driver from sending or receiving some messages.

18. If step 16 (reset attempt), is unsuccessful, then the CMI continues sending messages. The CMI does not attempt to reboot the system again without external intervention.

Note: A system that has locked up and can not be restarted with power button press is broken (bad power supply, short circuit on some bus, etc.)

The following rules apply if the system is in a G1 (S1 - S4) state:

- The CMI sends a message every period (30-32 seconds).
  - — If an event occurs prior to the system being shut down, the CMI immediately sends another event message with the next (incremented) sequence number.
  - — After the event, it resumes sending messages.

Note: There is a boundary condition when a hardware event happens right as the system transitions into a G0 state. In this condition, the hardware sends messages even though the system is in a G0 state (and the status bits could potentially indicate that). Normally the IICH does not send messages in the G0 state (except in the case of a lockup).

Note: A spurious alert could occur in the following sequence:

a. The CPU has initiated an alert using the SEND_NOW bit

b. During the alert, the PROCHOT#, INTRUDER# or GPI[11] changes state

c. The system then goes to a non-S0 state

Once the system transitions to the non-S0 state, it may send a single alert with an incremented SEQUENCE number.

Note: An inaccurate alert message can be generated in the following scenario:

a. The system successfully boots after a second watchdog Timeout occurs.

b. PWROK goes low (typically due to a reset button press) or a power button override occurs (before the SECOND_TO_STS bit is cleared).

c. An alert message indicating that the CPU is missing or locked up is generated with a new sequence number.

18.5.2 Message Format

The Event message is an SMBus Block Write sent to the External LAN Controller's SMBus address, as shown in Table 18-14.

Table 18-14. SMBus Message Format

S																															
1																															
Address Byte									**OpCode Byte**									**Length Byte**												
Address							**D**		**Opcode**									**Length**												
1	**1**	**0**	**0**	**1**	**0**	**0**	**0**		**1**		**0**	**0**	**1**	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**					
Data Byte 1									**Data Byte 2**									**Data Byte 3**												
(See Description)									**(See Description)**									**Sequence**				PwrSt		Rsvd						
CT	**BT**	**CS**	**Wt**	**SE**	**PE**	**FE**	**G**		**BB**	**BL**	**CF**	**0**	**0**	**0**	**0**	**0**		**B3**	B2	B1	B0	B1	B0	0	0					
Data Byte 4									**Data Byte 5**									**Data Byte 6**												
Message 1 Register									**Message 2 Register**									**WD Status Register**												
B7	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**		**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**		**B7**	B6	B5	B4	B3	B2	B1	B0					

Table 18-14. SMBus Message Format

Data Byte 7									Data Byte 8									
Reserved									Reserved									P
0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0		1

Table 18-15. Message Address Byte

Field	Bit Length	Comment
Start	1	'1' to indicate the start of a packet
Address	7	LAN SMBus Address. Is always 1100100.
Dir	1	'0' to indicate write cycle
Ack	1	Returned by External LAN Controller

Note: For the System Power State field: 00 = G0, 01 = G1, 10 = G2, 11=PreBoot. The preboot state is entered when the SLP_S3#, SLP_S4# and SLP_S5# signals go from low to high. The indication switches to the G0 state when CPURESET Done ACK completion packet sent to the IMCH. This corresponds to the time when the CPU has been reset. If the CPU is locked up, then the CPU EVENT bit is set.

18.5.3 Connecting an External LAN Controller

The CMI's TCO logic sends the message on the SMLINK signals in TCO compatible mode. An External LAN Controller claims these cycles.

When sending the messages to the external LAN Controller, the CMI's IICH abides by the standard SMBus rules associated with collision detection. It delays starting a message until the link is idle, and detects collisions. If a collision is detected, the CMI drops that message.

§ §

19.0 LPC Interface: Bus 0, Device 31, Function 0

19.1 Overview

The LPC bridge function IICH resides in PCI Device 31, Function 0. This contains many other functional units, such as DMA and Interrupt controllers, Timers, Power Management, System Management, GPIO, RTC, and LPC Configuration Registers.

19.2 LPC Interface Configuration Register Details

Note: Address locations that are not listed are considered reserved register locations. Reserved registers are read only and return all zeros. For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables".

Table 19-1. Bus 0, Device 31, Function 0: Summary of LPC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: ID: Vendor Identification Register" on page 734	50318086h
04h	05h	"Offset 04h: CMD: Device Command Register" on page 735	0007h
06h	07h	"Offset 06h: STS: Status Register" on page 736	0200h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 737	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 737	060100h
0Dh	0Dh	"Offset 0Dh: MLT: Master Latency Timer Register" on page 737	00h
0Eh	0Eh	"Offset 0Eh: HTYPE: Header Type Register" on page 738	80H
2Ch	2Fh	"Offset 2Ch: SID: Subsystem Identifiers Register" on page 738	00000000h
40h	43h	"Offset 40h: ABASE: ACPI Base Address Register" on page 739	00000001h
44h	47h	"Offset 44h: ACTL: ACPI Control Register" on page 739	00h
48h	48h	"Offset 48h: GBA: GPIO Base Address Register" on page 740	00000001h
4Ch	4Ch	"Offset 4Ch: GC: GPIO Control Register" on page 741	00h
60h	60h	"Offset 60h: PARC: PIRQA Routing Control Register" on page 741	80h
61h	61h	"Offset 61h: PBRC: PIRQB Routing Control Register" on page 742	80h
62h	62h	"Offset 62h: PCRC: PIRQC Routing Control Register" on page 742	80h
63h	63h	"Offset 63h: PDRC: PIRQDQ Routing Control Register" on page 743	80h
64h	64h	"Offset 64h: SCNT: Serial IRQ Control Register" on page 744	10h
68h	68h	"Offset 68h: PERC: PIRQEQ Routing Control Register" on page 745	80h
69h	69h	"Offset 69h: PFRC: PIRQF Routing Control Register" on page 745	80h
6Ah	6Ah	"Offset 6Ah: PGRC: PIRQG Routing Control Register" on page 746	80h

Table 19-1. Bus 0, Device 31, Function 0: Summary of LPC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
6Bh	6Bh	"Offset 6Bh: PHRC: PIRQH Routing Control Register" on page 747	80h
80h	81h	"Offset 80h: IOD: I/O Decode Ranges Register" on page 747	0000h
82h	83h	"Offset 82h: IOE: I/O Enables Register" on page 749	0000h
84h	85h	"Offset 84h: LG1: LPC Generic Decode Range 1 Register" on page 750	0000h
88h	88h	"Offset 88h: LG2: LPC Generic Decode Range 2 Register" on page 751	0000h
D0h	D3h	"Offset D0h: FS1: FWH ID Select 1 Register" on page 752	00112233h
D4h	D5h	"Offset D4h: FS2: FWH ID Select 2 Register" on page 753	4567h
D8h	DBh	"Offset D8h: FDE: FWH Decode Enable Register" on page 754	FFCFh
DCh	DCh	"Offset DCh: BC: BIOS Control Register" on page 756	00h
F0h	F3h	"Offset F0h: RCBA: Root Complex Base Address Register" on page 757	00000000h
F8h	FBh	"Offset F8h: MANID: Manufacturer ID Register" on page 757	00010F90h

19.2.1 PCI Configuration Registers

19.2.1.1 Offset 00h: ID: Vendor Identification Register

Table 19-2. Offset 00h: ID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** 50318086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	DID	**Device Identification:** These 16 bits of this register are hardwired to5031h.		5031h	RO
15 :00	VID	**Vendor Identification:** This 16-bit value is assigned to Intel. Intel VID = 8086h		8086h	RO

19.2.1.2 Offset 04h: CMD: Device Command Register

Table 19-3. Offset 04h: CMD: Device Command Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0007h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :10	Reserved	Reserved		00h	
09	FBE	**Fast Back to Back Enable:** Hardwired to '0' as per *PCI Express Specification*.		0h	RO
08	SEE	SERR# Enable: 0 = LPC bridge does not generates SERR# 1 = LPC bridge generates SERR#		0h	RW
07	WCC	**Wait Cycle Control:** Hardwired to '0' as per *PCI Express Specification*.		0h	RO
06	PERE	Parity Error Response Enable: 0 = No action is taken when detecting a parity error. 1 = Enables the LPC bridge to respond to parity errors detected on IICH interface.		0h	RW
05	VGA_PSE	**VGA Palette Snoop:** Hardwired to '0' as per *PCI Express Specification*.		0h	RO
04	MWIE	**Memory Write and Invalidate Enable:** Hardwired to '0'as per *PCI Express Specification*.		0h	RO
03	SCE	**Special Cycle Enable:** Hardwired to '0' as per *PCI Express Specification*.		0h	RO
02	BME	**Bus Master Enable:** Bus Masters cannot be disabled.		1	RO
01	MSE	**Memory Space Enable:** Memory space cannot be disabled on LPC.		1	RO
00	IOSE	**I/O Space Enable:** I/O space cannot be disabled on LPC.		1	RO

19.2.1.3 Offset 06h: STS: Status Register

Table 19-4. Offset 06h: STS: Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0200h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** Set when the LPC bridge detects an internal parity error. This bit gets set even if CMD.PERE is not set. 0 = Parity Error Not detected 1 = Parity Error detected		0h	RWC
14	SSE	**Signaled System Error:** Set when the LPC bridge signals a system error to the internal SERR# logic.		0h	RWC
13	RMA	**Received Master Abort:** 0 = Unsupported request status not received 1 = The bridge received a completion with unsupported request status from the IICH		0h	RWC
12	RTA	**Received Target Abort:** 0 = Completion abort not received 1 = Completion with completion abort received from the IICH		0h	RWC
11	STA	**Signaled Target Abort:** 0 = Target abort not generated on the IICH 1 = LPC bridge generated a completion packet with target abort status on the IICH		0h	RWC
10 :09	DTS	**DEVSEL# Timing Status:** 01 = Medium Timing		01h	RO
08	DPD	**Data Parity Error Detected:** 1 = 0 = All conditions listed below NOT met Set when all three of the following conditions are met: • LPC bridge receives a completion packet from the IICH from a previous request • Parity error has been detected (D31, F0, 06, bit 15) • PCICMD.PERE bit (D31, F0, 04, bit 6) is set		0h	RWC
07:00	Reserved	Reserved			

19.2.1.4 Offset 08h: RID - Revision ID Register

Writing to this register controls what is reported in all of the RID registers in the component. The value written does not get directly loaded in this register. However, the value is checked to determine which value to report.

Once written, additional writes to this register must not have any affect until a core-well reset occurs. BIOS must always write to this register in order to guarantee that the functionality is locked.

Table 19-5. Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	RID	**Revision ID:** Indicates the part revision		Variable	RWO

19.2.1.5 Offset 09h: CC: Class Code Register

Table 19-6. Offset 09h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 060100h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 :16	BCC	**Base Class Code:** indicates the type of device for the LPC bridge. 06h = Bridge device.		06h	RO
15 :08	SCC	**Sub-Class Code:** Indicates the category of bridge for the LPC bridge. 01h = PCI-to-ISA bridge.		01h	RO
07 :00	PI	**Programming Interface:** The LPC bridge has no programming interface.		00h	RO

19.2.1.6 Offset 0Dh: MLT: Master Latency Timer Register

Table 19-7. Offset 0Dh: MLT: Master Latency Timer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :03	MLC	**Master Latency Count:** Reserved per *PCI Express Specification*.		0h	RO
02 :00	Reserved	Reserved		0h	RO

19.2.1.7 Offset 0Eh: HTYPE: Header Type Register

Table 19-8. Offset 0Eh: HTYPE: Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 80H			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MFD	**Multi-function Device:** This bit is hardwired to '1' to indicate a multi-function device.		1	RO
06 :00	HTYPE	**Header Type:** Identifies the header layout of the configuration space, which is a generic device.		00h	RO

19.2.1.8 Offset 2Ch: SID: Subsystem Identifiers Register

This register is initialized to logic 0 by the assertion of PLTRST#. This register can be written only once after PLTRST# deassertion.

Table 19-9. Offset 2Ch: SID: Subsystem Identifiers Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	SSID	**Subsystem ID:** This is written by BIOS. No hardware action taken on this value.		0000h	RWO
15 :00	SSVID	**Subsystem Vendor ID:** This is written by BIOS. No hardware action taken on this value.		0000h	RWO

19.2.2 ACPI/GPIO Configuration Registers

19.2.2.1 Offset 40h: ABASE: ACPI Base Address Register

ABASE sets the base address in I/O space for the ACPI and TCO I/O registers (see Section 27.3.3, "General Power Management I/O-Mapped Registers" on page 1055). These registers can be mapped anywhere in the 64 K I/O space on 128-byte boundaries.

Table 19-10. Offset 40h: ABASE: ACPI Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 40h **Offset End:** 43h	
Size: 32bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved.		0000h	
15 :07	BSA	**Base Address**: This field provides the 128 bytes of I/O space for ACPI and TCO logic. This is placed on a 128 byte boundary.		00h	RW
06 :01	Reserved	Reserved.		00h	
00	RTE	**Resource Type Indicator**: Hardwired 1 to indicate I/O space.		1	RO

19.2.2.2 Offset 44h: ACT: ACPI Control Register

Table 19-11. Offset 44h: ACTL: ACPI Control Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 44h **Offset End:** 47h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	EN	ACPI Enable: 0 = Disable. Decoding of the I/O range pointed to by the ACPI base register is disabled, and the ACPI power management function is enabled. 1 = Decoding of the I/O range pointed to by the ACPI base register is enabled, and the ACPI power management function is enabled. ***Note:*** The APM power management ranges (B2/B3h) are always enabled and are not affected by this bit.		0h	RW
06 :03	Reserved	Reserved		0000h	

Table 19-11. Offset 44h: ACTL: ACPI Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 44h **Offset End:** 47h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02 :00	SCIS	**SCI IRQ Select:** Specifies on which IRQ the SCI will internally appear. If not using the APIC, the SCI must be routed to IRQ9-11, and that interrupt is not sharable with the SERIRQ stream, but is shareable with other PCI interrupts. If using the APIC, the SCI can also be mapped to IRQ20-23, and can be shared with other interrupts. Bits SCI Map 000 IRQ9 001 IRQ10 010 IRQ11 011 Reserved 100 RQ20 (only if APIC enabled) 101 IRQ21 (only if APIC enabled) 110 IRQ22 (only if APIC enabled) 111 IRQ23 (only if APIC enabled) ***Note:*** When the interrupt is mapped to APIC interrupts 9, 10 or 11, the APIC must be programmed for active-high reception. When the interrupt is mapped to APIC interrupts 20 through 23, the APIC must be programmed for active-low reception.		000h	RW

19.2.2.3 Offset 48h: GBA: GPIO Base Address Register

GBA sets the base address in I/O space for the GPIO I/O registers (see Section 22.2, "General Purpose I/O-Mapped Configuration Register Details" on page 806). These registers can be mapped anywhere in the 64 K I/O space on 64-byte boundaries.

Table 19-12. Offset 48h: GBA: GPIO Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 48h **Offset End:** 48h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved.		0000h	
15 :06	BA	**Base Address:** Provides the 64 bytes of I/O space for GPIO.		00h	RW
05 :01	Reserved	Reserved.		00h	
00	HD	Hardwired to 1 to indicate I/O space.		1	RO

19.2.2.4 Offset 4Ch: GC: GPIO Control Register

Table 19-13. Offset 4Ch: GC: GPIO Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 4Ch **Offset End:** 4Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :05	Reserved	Reserved.		000h	
04	EN	**GPIO Enable:** This bit enables/disables decode of the I/O range pointed to by the GPIO Base Address register (D31, F0, 48h) and enables the GPIO function. 0 = Disable 1 = Enable		0h	RW
03 :00	Reserved	Reserved.		0000h	

19.2.3 Interrupt Configuration Registers

19.2.3.1 Offset 60h: PARC: PIRQA Routing Control Register

Table 19-14. Offset 60h: PARC: PIRQA Routing Control Register

Description: PARC - Routing Control Register.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 60h **Offset End:** 60h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:*** BIOS must program this bit to 0 during POST for any of the PIRQs that are being used. The value of this bit may subsequently be changed by the OS when setting up for I/O APIC interrupt delivery mode.		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.2 Offset 61h: PBRC: PIRQB Routing Control Register

Table 19-15. Offset 61h: PBRC: PIRQB Routing Control Register

Description: PBRC - Routing Control Register.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 61h **Offset End:** 61h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	Interrupt Routing Enable: 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:*** BIOS must program this bit to 0 during POST for any of the PIRQs that are being used. The value of this bit may subsequently be changed by the OS when setting up for I/O APIC interrupt delivery mode.		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.3 Offset 62h: PCRC: PIRQC Routing Control Register

Table 19-16. Offset 62h: PCRC: PIRQC Routing Control Register (Sheet 1 of 2)

Description: PCRC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 62h **Offset End:** 62h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:*** BIOS must program this bit to 0 during POST for any of the PIRQs that are being used. The value of this bit may subsequently be changed by the OS when setting up for I/O APIC interrupt delivery mode.		1	RW
06 :04	Reserved	Reserved		000h	

Table 19-16. Offset 62h: PCRC: PIRQC Routing Control Register (Sheet 2 of 2)

Description: PCRC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 62h **Offset End:** 62h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.4 Offset 63h: PDRC: PIRQD Routing Control Register

Table 19-17. Offset 63h: PDRC: PIRQDQ Routing Control Register

Description: PDRC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 63h **Offset End:** 63h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:*** BIOS must program this bit to 0 during POST for any of the PIRQs that are being used. The value of this bit may subsequently be changed by the OS when setting up for I/O APIC interrupt delivery mode.		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.5 Offset 64h: SCNT: Serial IRQ Control Register

Table 19-18. Offset 64h: SCNT: Serial IRQ Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 64h **Offset End:** 64h	
Size: 8 bit	**Default:** 10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	EN	Enable: 0 = Serial IRQs will not be recognized 1 = Serial IRQs are recognized		0h	RW
06	MD	Mode: 0 = The serial IRQ machine is in quiet mode 1= The serial IRQ machine is in continuous mode ***Note:*** For systems using Quiet Mode, this bit must be set to 1 (Continuous Mode) for at least one frame after coming out of reset before switching back to Quiet Mode. Failure to do so will result in the IICH not recognizing SERIRQ interrupts.		0h	RW
05 :02	FS	**Frame Size:** 100 = Hardwired to indicate the size of the SERIRQ frame is 21 frames.		0100b	RO
01 :00	SFPW	**Start Frame Pulse Width:** This is the number of 33 MHz clocks that the SERIRQ pin is driven low by the Serial IRQ controller to signal a start frame. In continuous mode, the controller will drive the start frame for the number of clocks specified. In quiet mode, the controller will drive the start frame for the number of clocks specified minus one, as the first clock was driven by the peripheral. Bits Clocks 00 4 01 6 10 8 11 Reserved		00h	RW

When exiting S3/S4/S5, the following procedure must be used if the system needs Quiet Mode. Set the SERIRQ logic to continuous mode for at least one frame before switching it back to Quiet Mode.

19.2.3.6 Offset 68h: PERC: PIRQE Routing Control Register

Table 19-19. Offset 68h: PERC: PIRQEQ Routing Control Register

Description: PERC - Routing Control Register					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** 68h **Offset End:** 68h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:***		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.7 Offset 69h: PFRC: PIRQF Routing Control Register

Table 19-20. Offset 69h: PFRC: PIRQF Routing Control Register (Sheet 1 of 2)

Description: FARC - Routing Control Register					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** 69h **Offset End:** 69h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:***		1	RW
06 :04	Reserved	Reserved		000h	

Table 19-20. Offset 69h: PFRC: PIRQF Routing Control Register (Sheet 2 of 2)

Description: FARC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 69h **Offset End:** 69h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.8 Offset 6Ah: PGRC: PIRQG Routing Control Register

Table 19-21. Offset 6Ah: PGRC: PIRQG Routing Control Register

Description: PGRC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 6Ah **Offset End:** 6Ah	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:***		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.3.9 Offset 6Bh: PHRC: PIRQH Routing Control Register

Table 19-22. Offset 6Bh: PHRC: PIRQH Routing Control Register

Description: PHRC - Routing Control Register					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 6Bh **Offset End:** 6Bh	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	REN	**Interrupt Routing Enable:** 0 = The corresponding PIRQ is routed to one of the legacy interrupts specified in bits[03:00]. 1 = The PIRQ is not routed to the 8259. ***Note:***		1	RW
06 :04	Reserved	Reserved		000h	
03 :00	IR	IRQ Routing: Bits Mapping Bits Mapping 0000 Reserved 1000 Reserved 0001 Reserved 1001 IRQ9 0010 Reserved 1010 IRQ10 0011 IRQ3 1011 IRQ11 0100 IRQ4 1100 IRQ12 0101 IRQ5 1101 Reserved 0110 IRQ6 1110 IRQ14 0111 IRQ7 1111 IRQ15		0h	RW

19.2.4 LPC I/O Configuration Registers

19.2.4.1 Offset 80h: iOD: i/O Decode Ranges Register

Table 19-23. Offset 80h: IOD: I/O Decode Ranges Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 80h **Offset End:** 81h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :13	Reserved	Reserved.		0h	RO
12	FDD	**FDD Range:** This field determines which range to decode for the FDD Port 0 = 3F0 – 3F5h, 3F7h (Primary) 1 = 370 – 375h, 377h (Secondary)		0h	
11 :10	Reserved	Reserved		0h	RO

Table 19-23. Offset 80h: IOD: I/O Decode Ranges Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 80h **Offset End:** 81h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
9 :8	LPT	**LPT Range:** This field determines which range to decode for the LPT Port: 00 378 – 37Fh and 778 – 77Fh 01 278 – 27Fh (port 279h is read only) and 678 – 67Fh 10 3BC –3BEh and 7BC – 7BEh 11 Reserved		0h	
07	Reserved	Reserved		0h	RO
06 :04	CB	**ComB Range:** This field determines which range to decode for the COMB Port. 000 3F8 – 3FFh (COM1) 001 2F8 – 2FFh (COM2) 010 220 – 227h 011 228 – 22Fh 100 238 – 23Fh 101 2E8 – 2EFh (COM4) 110 338 – 33Fh 111 3E8 – 3EFh (COM3)		0h	
03	Reserved	Reserved		0h	RO
02 :00	CA	**ComA Range:** This field determines which range to decode for the COMA Port. 000 3F8 – 3FFh (COM1) 001 2F8 – 2FFh (COM2) 010 220 – 227h 011 228 – 22Fh 100 238 – 23Fh 101 2E8 – 2EFh (COM4) 110 338 – 33Fh 111 3E8 – 3EFh (COM3)		0h	

19.2.4.2 Offset 82h: IOE: I/O Enables Register

Table 19-24. Offset 82h: IOE: I/O Enables Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** 82h **Offset End:** 83h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :14	Reserved	Reserved		00h	
13	ME2	**Micro controller Enable 2:** 0 = Disable 1 = Enables the decoding of the I/O locations 4Eh and 4Fh to the LPC interface. This range is used for a microcontroller.		0h	RW
12	SE	**Super I/O Enable:** 0 = Disable 1 = Enables the decoding of the I/O locations 2Eh and 2Fh to the LPC interface. This range is used for Super I/O devices.		0h	RW
11	ME1	**Micro controller Enable 1:** 0 = Disable 1 = Enables the decoding of the I/O locations 62h and 66h to the LPC interface. This range is used for a microcontroller.		0h	RW
10	KE	**Keyboard Enable:** 0 = Disable 1 = Enables the decoding of the I/O locations 60h and 64h to the LPC interface. This range is used for a microcontroller.		0h	RW
09	HGE	High Gameport Enable: 0 = Disable 1 = Enables decoding of the I/O locations 208h to 20Fh to LPC		0h	RW
08	LGE	Low Gameport Enable: 0 = Disable 1 = Enables decoding of the I/O locations 200h to 207h to LPC		0h	RW
07 :04	Reserved	Reserved		0h	
03	FDE	Floppy Drive Enable: 0 = Disable 1 = Enables decoding of the FDD range to LPC. Range is selected by LIOD.FDE Decode Range Register (D31, F0, 80h, bit 12)		0h	RW

Table 19-24. Offset 82h: IOE: I/O Enables Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 82h **Offset End:** 83h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	PPE	Parallel Port Enable: 0 = Disable 1 = Enables decoding of the LPT range to LPC. Range is selected by LIOD.LPT Decode Range Register (D31, F0, 80h, bit 09:08)		0h	RW
01	CBE	Com Port B Enable: 0 = Disable 1 = Enables decoding of the COMB range to LPC. Range is selected LIOD.CB Decode Range Register (D31, F0, 80h, bits 06:04)		0h	RW
00	CAE	Com Port A Enable: 0 = Disable 1 = Enables decoding of the COMA range to LPC. Range is selected LIOD.CA Decode Range Register (D31, F0, 80h, bits 03:20)		0h	RW

19.2.4.3 Offset 84h: LG1: LPC Generic Decode Range 1 Register

LG1 sets the base address in I/O space for the I/O registers. These registers can be mapped anywhere in the 64 K I/O space on 128-byte boundaries.

Table 19-25. Offset 84h: LG1: LPC Generic Decode Range 1 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 84h **Offset End:** 85h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :07	BA	**Base Address:** Base Address for this generic decode range. This address is aligned on a 128-byte boundary, and being I/O, must have address lines 31:16 as 0. This generic decode is for I/O addresses only, not memory addresses. The size of this range is 128 bytes.		0h	RW
06 :01	Reserved	Reserved.		0h	
00	EN	Enable: 0 = Disable 1 = Enables the range specified in BA to be forwarded to LPC Interface		0h	RW

19.2.4.4 Offset 88h: LG2: LPC Generic Decode Range 2 Register

LG2 sets the base address in I/O space for the I/O registers. These registers can be mapped anywhere in the 64 K I/O space on 16-byte boundaries. The size of this region can be either 16B, 32B, or 64B based on the setting of ETR3.

Table 19-26. Offset 88h: LG2: LPC Generic Decode Range 2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** 88h **Offset End:** 88h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :04	BA	**Base Address:** This address is aligned on a 16-byte boundary, and must have address lines 31:16 as 0. Note that configuration bits at D31:F0:ACh (bits 13 and 12) allow this range to be increased to 32 or 64 bytes by forcing matches on address bits 4 and/or 5.		0h	RW
03 :01	Reserved	Reserved.		0h	
00	EN	Enable: 0 = Disable 1 = Enables the range specified in BA to be forwarded to LPC Interface		0h	RW

19.2.5 Power Management Configuration Registers

Offsets A0h – CFh are described in the power management section. Refer to Chapter 27.0, “Power Management”.

19.2.6 FWH Configuration Registers

19.2.6.1 Offset D0h: FS1: FWH ID Select 1 Register

This register contains the IDSEL fields the LPC Bridge uses for memory cycles going to the FWH.

Table 19-27. Offset D0h: FS1: FWH ID Select 1 Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** D0h **Offset End:** D3h	
Size: 32 bit	**Default:** 00112233h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	IF8	**F8-FF IDSEL:** Is used in FWH cycle for ranges enabled by FDE.EF8. Used for two 512-KB Firmware Hub memory ranges and one 128-KB memory range. This field is fixed at 0000. The IDSEL programmed in this field addresses the following memory ranges: FFF8 0000h – FFFF FFFFh FFB8 0000h – FFBF FFFFh 000E 0000h – 000F FFFFh		0h	RO
27 :24	IF0	**F0-F7 IDSEL:** Is used in FWH cycle for range enabled by FDE.EF0. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFF0 0000h – FFF7 FFFFh FFB0 0000h – FFB7 FFFFh		0h	RW
23 :20	IE8	**E8-EF IDSEL:** Is used in FWH cycle for range enabled by FDE.EE8. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFE8 0000h – FFEF FFFFh FFA8 0000h – FFAF FFFFh		1h	RW
19 :16	IE0	**E0-E7 IDSEL:** Is used in FWH cycle for range enabled by FDE.EE0. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFE0 0000h – FFE7 FFFFh FFA0 0000h – FFA7 FFFFh		1h	RW
15 :12	ID8	**D8-DF IDSEL:** Is used in FWH cycle for range enabled by FDE.ED8. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFD8 0000h – FFDF FFFFh FF98 0000h – FF9F FFFFh		2h	RW
11 :08	ID0	**D0-D7 IDSEL:** Is used in FWH cycle for range enabled by FDE.ED0. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFD0 0000h – FFD7 FFFFh FF90 0000h – FF97 FFFFh		2h	RW
07 :04	IC8	**C8-CF IDSEL:** Is used in FWH cycle for range enabled by FDE.EC8. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFC8 0000h – FFCF FFFFh FF88 0000h – FF8F FFFFh		3h	RW
03 :00	IC0	**C0-C7 IDSEL:** Is used in FWH cycle for range enabled by FDE.EC0. Used for two 512-KB Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FFC0 0000h – FFC7 FFFFh FF80 0000h – FF87 FFFFh		3h	RW

19.2.6.2 Offset D4h: FS2: FWH ID Select 2 Register

This register contains the additional IDSEL fields the LPC Bridge uses for memory cycles going to the FWH.

Table 19-28. Offset D4h: FS2: FWH ID Select 2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** D4h **Offset End:** D5h	
Size: 16 bit	**Default:** 4567h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :12	I70	**70-7F IDSEL:** Is used in FWH cycle for range enabled by FDE.E70. Used for two, 1-M Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FF70 0000h – FF7F FFFFh FF30 0000h – FF3F FFFFh		4h	RW
11 :08	I60	**60-6F IDSEL:** Is used in FWH cycle for range enabled by FDE.E60. Used for two, 1-M Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FF60 0000h – FF6F FFFFh FF20 0000h – FF2F FFFFh		5h	RW
07 :04	I50	**50-5F IDSEL:** Is used in FWH cycle for range enabled by FDE.E50. Used for two, 1-M Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FF50 0000h – FF5F FFFFh FF10 0000h – FF1F FFFFh		6h	RW
03 :00	I40	**40-4F IDSEL:** Is used in FWH cycle for range enabled by FDE.E40. Used for two, 1-M Firmware Hub memory ranges. The IDSEL programmed in this field addresses the following memory ranges: FF40 0000h – FF4F FFFFh FF00 0000h – FF0F FFFFh		7h	RW

19.2.6.3 Offset D8h: FDE: FWH Decode Enable Register

Table 19-29. Offset D8h: FDE: FWH Decode Enable Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** D8h **Offset End:** DBh	
Size: 16 bit	**Default:** FFCFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	EF8	**F8-FF Enable:** Enables decoding of 512 Kbyte Firmware Hub memory ranges: 0 = 0 = Disable 1 = 1 = Enable the following ranges for the Firmware Hub Data space:FFF80000h – FFFFFFFFh Feature space:FFB80000h – FFBFFFFFh		1	RO
14	EF0	**F0-F8 Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFF00000h – FFF7FFFFh Feature space:FFB00000h – FFB7FFFFh		1	RW
13	EE8	**E8-EF Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFE80000h – FFEFFFFFh Feature space:FFA80000h – FFAFFFFFh		1	RW
12	EE0	**E0-E8 Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFE00000h – FFE7FFFFh Feature Space:FFA00000h – FFA7FFFFh		1	RW
11	ED8	**D8-DF Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFD80000h – FFDFFFFFh Feature space:FF980000h – FF9FFFFFh		1	RW
10	ED0	**D0-D7 Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFD00000h – FFD7FFFFh Feature space:FF900000h – FF97FFFFh		1	RW
09	EC8	**C8-CF Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFC80000h – FFCFFFFFh Feature space:FF880000h – FF8FFFFFh		1	RW

Table 19-29. Offset D8h: FDE: FWH Decode Enable Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:0		**Offset Start:** D8h **Offset End:** DBh	
Size: 16 bit	**Default:** FFCFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	EC0	**C0-C7 Enable:** Enables decoding of 512K-Kbyte Firmware Hub memory ranges: 0 = Disable. 1 = Enable the following ranges for the Firmware Hub: Data space:FFC00000h – FFC7FFFFh Feature space:FF800000h – FF87FFFFh		1	RW
07	LFE	**Legacy F Segment Enable:** This enables the decoding of the legacy 128K range at F0000h – FFFFFh 0 = Disable. 1 = Enable the following legacy ranges for the Firmware Hub F0000h – EFFFFh		1	RW
06	LEE	**Legacy E Segment Enable:** This enables the decoding of the legacy 128K range at E0000h – EFFFFh 0 = Disable. 1 = Enable the following legacy ranges for the Firmware Hub E0000h – EFFFFh		1	RW
05 :04	Reserved	Reserved		00h	
03	E70	**70-7F Enable:** Enables decoding of 1 MByte Firmware Hub memory range: Data space:FF700000h – FF7FFFFFh Feature space:FF300000h – FF3FFFFFh		1	RW
02	E60	**60-6F Enable:** Enables decoding of 1 MByte Firmware Hub memory range: Data space:FF600000h – FF6FFFFFh Feature Space:FF200000h – FF2FFFFFh		1	RW
01	E50	**50-5F Enable:** Enables decoding of 1 MByte Firmware Hub memory range: Data space:FF500000h – FF5FFFFFh Feature space:FF100000h – FF1FFFFFh		1	RW
00	E40	**40-4F Enable:** Enables decoding of 1 MByte Firmware Hub memory range: Data space:FF400000h – FF4FFFFFh Feature space:FF0000000h – FF0FFFFFh		1	RW

19.2.6.4 Offset DCh: BC: BIOS Control Register

Table 19-30. Offset DCh: BC: BIOS Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :04	Reserved	Reserved		0h	
03 02	SRC	**SPI Read Configuration:** This 2-bit field controls two policies related to BIOS reads on the SPI interface Bit 3 - Prefetch Enable Bit 2 - Cache Disable **00b -> No prefetching, but caching enabled**. 64B demand reads load the read buffer cache with "valid" data, allowing repeated code fetches to the same line to complete quickly **01b -> No prefetching and no caching.** One-to-one correspondence of the host BIOS reads to SPI cycles. This value can be used to invalidate the cache. **10b -> Prefetching and Caching enabled.** This mode is used for long sequences of short reads to consecutive addresses (i.e. shadowing) **11b -> Reserved**. This is an invalid configuration.		00h	RW
01	LE	**Lock Enable:** 0 = Setting the WP will not cause SMIs 1 = Enables setting the WP bit to cause SMIs. Once set, this bit can only be cleared by a PLTRST#		0h	RWO
00	WP	**Write Protect:** 0 = Only read cycles result in Firmware Hub Interface cycles 1 = Access to the BIOS space is enabled for both read and write cycles. When this bit is written from a 0 to a 1 and Lock Enable (LE) is also set, an SMI# is generated. This ensures that only SMI code can update BIOS.		0h	RW

19.2.7 Root Complex Register Block Configuration Register

19.2.7.1 Offset F0h: RCBA: Root Complex Base Address Register

RCBA sets the base address in memory space for the Root Complex Configuration registers (see Section 17.1, "Root Complex Memory-Mapped Configuration Register Details" on page 689). These registers can be mapped anywhere in the 32-bit memory space on 16KB boundaries.

The SPI register space resides in this BAR. The base address offset is 3020h. Refer to Chapter 21.0, "Serial Peripheral Interface".

Table 19-31. Offset F0h: RCBA: Root Complex Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** F0h **Offset End:** F3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :14	BA	**Base Address:** Base Address for the root complex register block decode range. This address is aligned on a 16 Kbyte boundary.		0h	RW
13 :01	Reserved	Reserved		0h	
00	EN	Enable: 0 = Disables the range specified in BA to be claimed as the RCRB (Root Complex Register Block) 1 = Enables the range specified in BA to be claimed as the RCRB		0h	RW

19.2.8 Manufacturing Information Register

19.2.8.1 Offset F8h: MANID: Manufacturer ID Register

Table 19-32. Offset F8h: MANID: Manufacturer ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:** 00010F90h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved		00h	
23 :16	SID	**Stepping Identifier:** This field increments for each stepping of the part. ***Note:*** This field can be used by software to differentiate steppings when the Revision ID may not change. ***Note:*** 00h for A0 stepping ***Note:*** 01h for B0 stepping A single Stepping ID can be implemented that is readable from all functions in the chip because all of them increment in lock-step.		01h	RO
15 :08	MID	**Manufacturing Identifier:** 0Fh = Intel		0Fh	RO
07 :00	Reserved	Reserved.		90h	

19.3 Interface

The LPC bridge function resides in Device 31, Function 0. In addition to the LPC bridge function, D31, F0 contains other functional units including DMA, Interrupt controllers, Timers, Power Management, System Management, GPIO, and RTC.

19.3.1 Overview

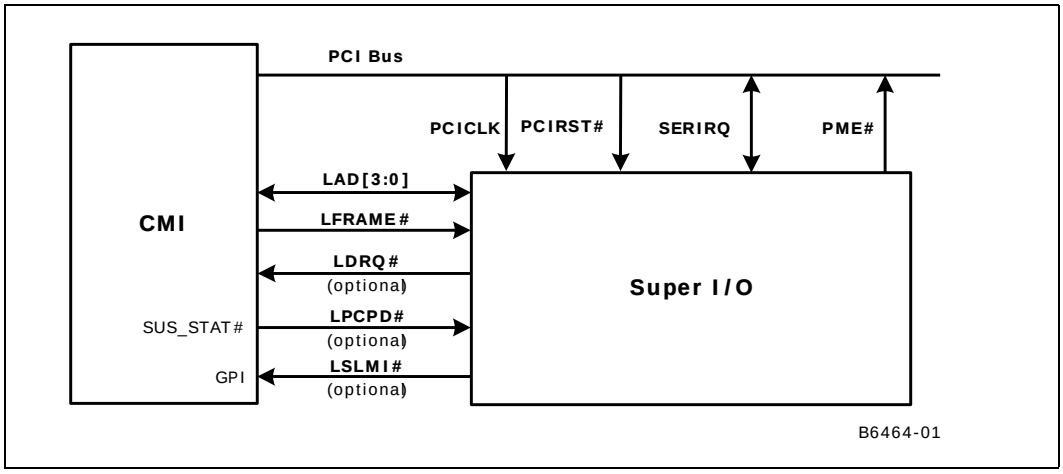
The LPC interface is described in the *Low Pin Count (LPC) Interface Specification, Revision 1.1*. The LPC interface to the IICH is shown in Figure 19-1. The LPC Controller implements all of the signals that are shown as optional, but peripherals are not required to do so.

For the LPC Controller:

- LSMI# can be connected to any of the SMI capable GPIO signals.
- The Super I/O's PME# can be connected to the PCI PME# signal, however this may cause software problems. A better choice is to connect it to one of the LPC Controller's SCI capable GPIO signals.
- The LPC Controller's SUS_STAT# signal is connected directly to the LPCPD# signal.

All the other signals have the same name on the LPC Controller and on the LPC Interface.

Figure 19-1. LPC Interface Diagram



Signal Name	During Reset	After Reset		S3	S5
LAD[3:0]	See			Off	Off
LFRAME#				Off	Off
LDREQ[1:0]#				Off	Off

19.3.2 Cycle Types

All of the cycle types implemented are described in the *LPC Interface Specification, Revision 1.1*.Table 19-33 shows the supported cycle types.

Table 19-33. LPC Cycle Types Supported

Cycle Type	Comment
Memory Read	Single: 1 byte only
Memory Write	Single: 1 byte only
I/O Read	1 byte only. breaks up 16- and 32-bit processor cycles into multiple 8-bit transfers. See Note 1 below.
I/O Write	1 byte only. breaks up 16- and 32-bit processor cycles into multiple 8-bit transfers. See Note 1 below.
DMA Read	Can be 1, or 2 bytes
DMA Write	Can be 1, or 2 bytes
Bus Master Read	Can be 1, 2, or 4 bytes. (See Note 2 below)
Bus Master Write	Can be 1, 2, or 4 bytes. (See Note 2 below)

Notes:

1. For memory cycles below 16 MB that do not target enabled firmware hub ranges, performs standard LPC memory cycles. It only attempts 8-bit transfers. If the cycle appears on PCI as a 16-bit transfer, it appears as two consecutive 8-bit transfers on LPC. Likewise, if the cycle appears as a 32-bit transfer on PCI, it appears as four consecutive 8-bit transfers on LPC. If the cycle is not claimed by any peripheral, it is subsequently aborted, and returns a value of all ones to the processor. This is done to maintain compatibility with legacy memory cycles where pull-up resistors would keep the bus high if no device responds.
2. Bus Master Read or Write cycles must be naturally aligned. For example, a 1-byte transfer can be to any address. However, the 2-byte transfer must be word aligned (i.e., with an address where A0=0). A DWord transfer must be DWord aligned (i.e., with an address where A1 and A0 are both 0).

The *Low Pin Count (LPC) Interface Specification, Revision 1.1* allows DMA cycles to be 4-bytes in length, but the LPC controller will only allow a maximum of 16-bit transfers. Additionally, the *LPC Specification* allows for firmware memory cycles to be 1, 2, or 4 bytes, and in the case of firmware reads, 128 bytes. However, the LPC controller will only perform 8-bit transfers.

Bus master read or write cycles must be naturally aligned. A 1 byte transfer can be to any address. A 2-byte transfer must be word aligned (address bit A0 = 0). A 4-byte transfer must be Dword aligned (address bits A[1:0] = 00).

19.3.3 Aborting a Cycle

The usage of LFRAME# is followed as it is defined in the *LPC Specification*.

The LPC Controller performs an abort for the following cases (possible failure cases):

- LPC Controller starts a Memory, I/O, or DMA cycle, but no device drives a valid SYNC after four consecutive clocks.
- LPC Controller starts a Memory, I/O, or DMA cycle, and the peripheral drives an invalid SYNC pattern.
- A peripheral drives an illegal address when performing bus master cycles.
- A peripheral drives an invalid value.

19.3.4 Memory Cycle Notes

For cycles below 16M and not targeting firmware, the LPC Controller will perform standard LPC memory cycles. For cycles targeting firmware, firmware memory cycles are used. For cycles targeting the fixed token, the fixed token format is used. Only 8-bit transfers are performed. If a larger transfer occurs, the LPC controller will break it into multiple 8-bit transfers until the request is satisfied.

Note: If the cycle is not claimed by any peripheral (and subsequently aborted), a value of all 1s (FFh) is returned to the processor. This is to maintain compatibility with legacy memory cycles where pull-up resistors would keep the bus high if no device responds.

19.3.5 I/O Cycle Notes

For I/O cycles targeting registers specified in the decode ranges, CMI performs I/O cycles as defined in the *LPC Specification*. These are 8-bit transfers. If the processor attempts a 16-bit or 32-bit transfer, CMI breaks the cycle up into multiple 8-bit transfers to consecutive I/O addresses until the request is satisfied.

Note: If the cycle is not claimed by any peripheral (and subsequently aborted), CMI returns a value of all 1s (FFh) to the processor. This is to maintain compatibility with legacy I/O cycles where pull-up resistors would keep the bus high if no device responds.

19.3.6 DMA Cycle Notes

Only 8-bit and 16-bit DMA transfers are supported. Peripherals must not attempt 32-bit transfers.

19.3.7 Bus Master Cycle Notes

CMI supports Bus Master cycles and requests (using LDRQ#) as defined in the *LPC Specification*. CMI has two LDRQ# inputs, and thus supports two separate bus master devices. It uses the associated START fields for Bus Master 0 (0010b) or Bus Master 1 (0011b).

Note: CMI does not support LPC Bus Masters performing I/O cycles. LPC Bus Masters must only perform memory read or memory write cycles and must only target main memory.

19.3.8 FWH Cycle Notes

A FWH device is not allowed to assert an Error SYNC. If the LPC controller receives any SYNC returned from the device other than short wait (0101), long wait (0110), or ready more (0000) when running a FWH cycle, indeterminate results will occur.

19.3.9 LPC PD# Protocol

The LPCPD# pin must follow the same timings as for SUS_STAT#.Upon driving SUS_STAT# low, LPC peripherals will drive LDRQ# low or tri-state it. The LPC Controller must shut the LDRQ# input buffers. After driving SUS_STAT# active, the LPC Controller drives LFRAME# low, and tri-states (or drive low) LAD[3:0].

Note: The LPC Controller does not follow one part of the *LPC Specification* that says “LRESET# is always asserted after LPCPD#”. LRESET# is not always asserted after LPCPD#. The *Low Pin Count Interface Specification, Revision 1.1* defines the LPCPD# protocol where there is at least 30 µs from LPCPD# assertion to LRST# assertion. This specification explicitly states that this protocol only applies to entry/exit of low power states which does not include asynchronous reset events. CMI asserts both SUS_STAT# (connects to LPCPD#) and PLTRST# (connects to LRST#) at the same time when the core logic is reset (via CF9h, PWROK, or SYS_RESET#, etc.). This is not inconsistent with the LPC LPCPD# protocol.

19.3.10 Cycle Posting Policies

Three main policies are assumed.

1. I/O cycles and memory read cycles from the processor are not posted. Memory write Cycles from the processor are posted.
2. DMA cycles can be pipelined. For example, after reading data from memory, the LPC Controller can then release PHOLD# while it writes the data to the peripheral on the LPC Interface. This is because there are no processor/SMI#-based retry capabilities for DMA cycles. In the other direction, after reading data from a peripheral, PHOLD# can be released while the DMA controller writes data to main memory.
3. When bus masters read from main memory, the LPC Controller can treat this much like DMA, and release the memory and PCI buses while the data is being transferred to the bus master on the LPC Interface. When a bus master writes to main memory, the LPC Controller can use the LPC Interface while the data is being written to main memory.

19.3.11 Configuration

19.3.11.1 LPC Interface Decoders

To allow the I/O cycles and memory mapped cycles to go to the LPC interface, CMI includes several decoders. During configuration, CMI must be programmed with the same decode ranges as the peripheral. The decoders are programmed via the Device 31, Function 0 configuration space at offset 80h–87h.

Note: CMI cannot accept PCI write cycles from PCI-to-PCI bridges or devices with similar characteristics (specifically those with a “Retry Read” feature which is enabled) to an LPC device if there is an outstanding LPC read cycle towards the same PCI device or bridge. These cycles are not part of normal system operation, but may be encountered as part of platform validation testing using custom test fixtures.

19.3.11.2 Bus Master Device Mapping and START Fields

Bus Masters must have a unique START field. In the case of the LPC Controller, which supports two bus masters, it will drive 0010 for the START field for grants to bus master 0 (requested via LDRQ[0]#) and 0011 for grants to bus master 1 (requested via LDRQ[1]#.).

19.3.11.3 Firmware Memory IDSEL fields

The LPC Controller uses a unique IDSEL field for each EPROM. The IDSEL used is determined through the programming of the FS1 and FS2 configuration registers.

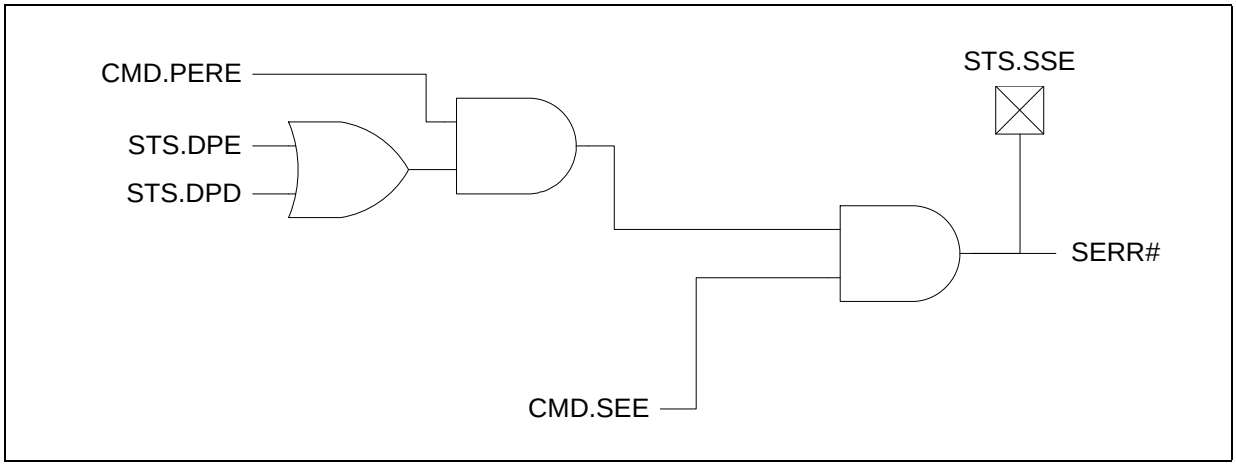
19.3.12 SERR# Generation

Several internal and external sources of the LPC Bridge can cause SERR#, and are described below.

The first class of errors is parity errors related internally to CMI. The LPC Bridge captures generic data parity errors (errors it finds internally), as well as, errors returned on internal cycles where the bridge was the master. If either of these two conditions are met, and the bridge is enabled for parity error response, SERR# is captured.

Additionally, if the bridge receives a target abort or master abort, and the bridge policy is to SERR# on these types of aborts, SERR# is generated.

Figure 19-2. LPC Bridge SERR#



CMD.PERE - Offset 04-05h bit 06

STS.DPE - Offset 06-07h bit 15

STS.DPD - Offset 06-07h bit 08

STS.RTA - Offset 06-07h bit 12

STS.RMA - Offset 06-07h bit 13

§ §

20.0 LPC DMA

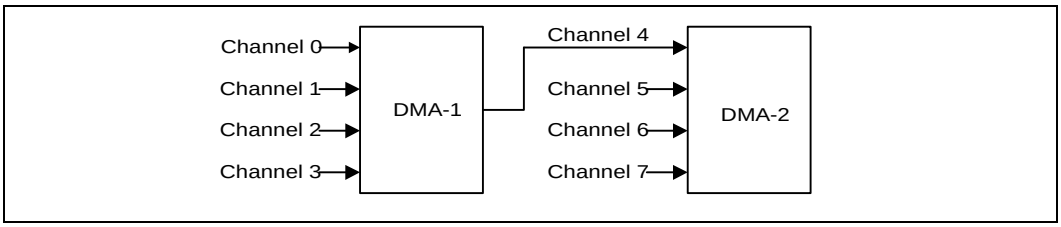
20.1 Overview

LPC DMA is supported using the IICH DMA controller. The DMA controller has registers that are fixed in the lower 64 Kbyte of I/O space. The DMA controller is configured using registers in the PCI configuration space. These registers allow configuration of the channels for use by LPC DMA.

The DMA circuitry incorporates the functionality of two 8237 DMA controllers with seven independently programmable channels; Channels 0–3 and Channels 5–7. DMA Channel 4 cascades the two controllers together and is not available for other use.

In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that software initiates. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to a one. The DMA controller is used for LPC DMA.

Figure 20-1. IICH DMA Controller



Feature set of the IICH DMA controller:

- Channels 0–3 provide 8-bit, count-by-bytes transfers.
- Channels 5–7 provide 16-bit, count-by-words transfers.
- 24-bit addressing. Each channel includes a 16-bit, legacy-compatible Current Address Register (CAR), which holds the 16 least-significant bits, and an legacy Compatible Page Register, which contains the eight next most significant bits of address.
- Auto-initialization following a DMA termination.

The DMA controller has registers that are fixed in the lower 64 Kbyte of I/O space.

The DMA controller is configured using registers in the PCI configuration space. These registers allow configuration of individual channels for use by LPC.

The RPR bit effects the register decoding for the Reserved Page register (Addresses 80, 84-86, 88, 8C-8E and their aliases in the 9x range). See Section 17.1.6.3, “Offset 3410h: GCS: General Control and Status Register”bit 2.

20.2 LPC DMA I/O-Mapped Register Details

Note: Some registers are normally read-only, but are writable in Alt-Access mode. Likewise, there are some registers that are normally write-only, but are readable in Alt-Access mode. The individual register descriptions may not indicate this. See Alt-Access mode for more details.

Table 20-1. Summary of LPC DMA Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
00h at 02h	10h at 02h	"Offset 00h: DMA_BCA[0-3] - DMA Base and Current Address Registers for Channels 0-3" on page 766	XXXX
C4h at 04h	C5h at 04h	"Offset C4h: DMA_BCA[5-7] - DMA Base and Current Address Registers for Channels 5-7" on page 767	XXXX
01h at 02h	11h at 02h	"Offset 01h: DMA_BCC[0-3] - DMA Base and Current Count Registers for Channels 0-3" on page 768	XXXX
C6h at 04h	C7h at 04h	"Offset C6h: DMA_BCC[5-7] - DMA Base and Current Count Registers for Channels 5-7" on page 769	XXXX
87h, 83h, 81h, 82h	97h, 93h, 91h, 82h	"Offset 87h: DMA_MPL[0-3] - DMA Memory Low Page Registers for Channels 0-3" on page 771	XXXXXXX
8Bh, 89h, 8Ah	9Bh, 99h, 9Ah	"Offset 8Bh: DMA_MPL[5-7]: DMA Memory Low Page Registers for Channels 5-7" on page 771	XXXXXXX

Table 20-2. 0000h (IO) Base Address Registers in the IA F1 View

Offset Start	Offset End	Register ID - Description	Default Value
08h	08h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
18h	18h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
08h	08h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
18h	18h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
0Ah	0Ah	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
1Ah	1Ah	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
0Bh	0Bh	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
1Bh	1Bh	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
0Ch	0Ch	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
1Ch	1Ch	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
0Dh	0Dh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
1Dh	1Dh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
0Eh	0Eh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
1Eh	1Eh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
0Fh	0Fh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b
1Fh	1Fh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b

Table 20-3. 0000h (IO) Base Address Registers in the IA F2 View

Offset Start	Offset End	Register ID - Description	Default Value
D0h	D0h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
D1h	D1h	"Offset 08h: DMA_COMMAND - DMA Command Register" on page 770	000X0X00b
D0h	D0h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
D1h	D1h	"Offset 08h: DMA_STATUS - DMA Status Register" on page 772	XXXXXXXh
D4h	D4h	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
D5h	D5h	"Offset 0Ah: DMA_WSM - DMA Write Single Mask Register" on page 773	000001xxb
D6h	D6h	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
D7h	D7h	"Offset 0Bh: DMA_CHM - DMA Channel Mode Register" on page 774	000000XXh
D8h	D8h	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
D9h	D9h	"Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register" on page 775	XXXXXXXXh
DAh	DAh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
DBh	DBh	"Offset 0Dh: DMA_MC - DMA Master Clear Register" on page 775	XXXXXXXXh
DCh	DCh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
DDh	DDh	"Offset 0Eh: DMA_CM - DMA Clear Mask Register" on page 776	XXXXXXXXh
DEh	DEh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b
DFh	DFh	"Offset 0Fh: DMA_WAM - DMA Write All Mask Register" on page 777	00001111b

20.2.1 Register Descriptions

20.2.1.1 Offset 00h: DMA_BCA[0-3] - DMA Base and Current Address Registers for Channels 0-3

Table 20-4. Offset 00h: DMA_BCA[0-3] - DMA Base and Current Address Registers for Channels 0-3

Description:	Ch. 0: 00h - 10h; Ch.1: 02h - 12h; Ch.2: 04h - 14h; Ch.3: 06h - 16h				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 00h at 02h **Offset End:** 10h at 02h	
Size: 16 bit[a]	**Default:** XXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	BCADD	**Base and Current Address:** This register determines the address for the transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the Base Address register and copied to the Current Address register. On reads, the value is returned from the Current Address register. The address increments/decrements in the Current Address register after each transfer, depending on the mode of the transfer. If the channel is in auto-initialize mode, the Current Address register is reloaded from the Base Address register after a terminal count is generated. For transfers to/from a 16-bit slave (channels 5-7), the address is shifted left one bit location. Bit 15 is shifted into bit 16. The register is accessed in 8-bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing an address register, the byte pointer flip/flop must be cleared to ensure that the low byte is accessed first.		XXXX	RW

a. This register provides an 8-bit window into a 16-bit quantity. The byte accessed depends on the current byte pointer flip/flop.

20.2.1.2 Offset C4h: DMA_BCA[5-7] - DMA Base and Current Address Registers for Channels 5-7

Table 20-5. Offset C4h: DMA_BCA[5-7] - DMA Base and Current Address Registers for Channels 5-7

Description:	Ch. 5: C4h - C5h; Ch. 6: C8h - C9h; Ch. 7: CCh - CDh				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** C4h at 04h **Offset End:** C5h at 04h	
Size: 16 bit[a]	**Default:** XXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	BCADD	**Base and Current Address:** This register determines the address for the transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the Base Address register and copied to the Current Address register. On reads, the value is returned from the Current Address register. The address increments/decrements in the Current Address register after each transfer, depending on the mode of the transfer. If the channel is in auto-initialize mode, the Current Address register is reloaded from the Base Address register after a terminal count is generated. For transfers to/from a 16-bit slave (channels 5-7), the address is shifted left one bit location. Bit 15 is shifted into bit 16. The register is accessed in 8-bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing an address register, the byte pointer flip/flop must be cleared to ensure that the low byte is accessed first.		XXXXh	RW

a. This register provides an 8-bit window into a 16-bit quantity. The byte accessed depends on the current byte pointer flip/flop.

20.2.1.3 Offset 01h: DMA_BCC[0-3] - DMA Base and Current Count Registers for Channels 0-3

Table 20-6. Offset 01h: DMA_BCC[0-3] - DMA Base and Current Count Registers for Channels 0-3

Description:	Ch. 0: 01h - 11h; Ch. 1: 03h - 13h; Ch. 2: 05h - 15h; Ch. 3: 07h - 17h,				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 01h at 02h **Offset End:** 11h at 02h	
Size: 16 bit[a]	**Default:** XXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	BCCNT	**Base and Current Count:** This register determines the number of transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the Base Count register and copied to the Current Count register. On reads, the value is returned from the Current Count register. The actual number of transfers is one more than the number programmed in the Base Count Register (i.e., programming a count of 4h results in 5 transfers). The count is decremented in the Current Count register after each transfer. When the value in the register rolls from zero to FFFFh, a terminal count is generated. If the channel is in auto-initialize mode, the Current Count register is reloaded from the Base Count register after a terminal count is generated. For transfers to/from an 8-bit slave (channels 0-3), the count register indicates the number of bytes to be transferred. For transfers to/from a 16-bit slave (channels 5-7), the count register indicates the number of words to be transferred. The register is accessed in 8-bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing a count register, the byte pointer flip/flop must be cleared to ensure that the lowest byte is accessed first.		XXXXh	RW

a. This register provides an 8-bit window into a 16-bit quantity. The byte accessed depends on the current byte pointer flip/flop.

20.2.1.4 Offset C6h: DMA_BCC[5-7] - DMA Base and Current Count Registers for Channels 5-7

Table 20-7. Offset C6h: DMA_BCC[5-7] - DMA Base and Current Count Registers for Channels 5-7

Description:	Ch. 5: C6h - C7h, Ch. 6: CAh - CBh,Ch. 7: CEh - CFh				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** C6h at 04h **Offset End:** C7h at 04h	
Size: 16 bit[a]	**Default:** XXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	BCCNT	**Base and Current Count:** This register determines the number of transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the Base Count register and copied to the Current Count register. On reads, the value is returned from the Current Count register. The actual number of transfers is one more than the number programmed in the Base Count Register (i.e., programming a count of 4h results in 5 transfers). The count is decremented in the Current Count register after each transfer. When the value in the register rolls from zero to FFFFh, a terminal count is generated. If the channel is in auto-initialize mode, the Current Count register is reloaded from the Base Count register after a terminal count is generated. For transfers to/from an 8-bit slave (channels 0-3), the count register indicates the number of bytes to be transferred. For transfers to/from a 16-bit slave (channels 5-7), the count register indicates the number of words to be transferred. The register is accessed in 8-bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing a count register, the byte pointer flip/flop must be cleared to ensure that the lowest byte is accessed first.		XXXXh	RW

a. This register provides an 8-bit window into a 16-bit quantity. The byte accessed depends on the current byte pointer flip/flop.

20.2.1.5 Offset 08h: DMA_COMMAND - DMA Command Register

Table 20-8. Offset 08h: DMA_COMMAND - DMA Command Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 08h **Offset End:** 08h	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 18h **Offset End:** 18h	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** D0h **Offset End:** D0h	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** D1h **Offset End:** D1h	
Size: 8 bit	**Default:** 000X0X00b			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	Reserved	Reserved. Software must always write 0 to these bits.		0h	
04	DMAARB	**DMA Group Arbitration Priority:** Each channel group is individually assigned either fixed or rotating arbitration priority. At reset, each group is initialized in fixed priority. 0 = Assigns fixed priority to the channel group. 1 = Assigns rotating priority to the channel group.		Xh	WO
03	Reserved	Reserved. Must be 0		0h	
02	DMACGEN	**DMA Channel Group Enable:** Both channel groups are enabled following part reset. Disabling channel group 4-7 also disables channel group 0-3, which is cascaded through channel 4. 0 = Enables the DMA channel group 1 = Disables the DMA channel group		X	WO
01 : 00	Reserved	Reserved. Must be 0.		0h	

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.6 Offset 87h: DMA_MPL[0-3] - DMA Memory Low Page Registers for Channels 0-3

Table 20-9. Offset 87h: DMA_MPL[0-3] - DMA Memory Low Page Registers for Channels 0-3

Description:	Ch. 0: 87h - 97h, Ch. 1: 83h - 93h, Ch. 2: 81h - 91h, Ch. 3: 82h,				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 87h, 83h, 81h, 82h **Offset End:** 97h, 93h, 91h, 82h	
Size: 8 bit	**Default:** XXXXXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DMALP	**DMA Low Page (ISA Address bits [23:16]):** This register works in conjunction with the DMA controller's Current Address Register to define the complete 24-bit address for the DMA channel. This register remains static throughout the DMA transfer. Bit 16 of this register is ignored when in 16 bit I/O count by words mode as it is replaced by the bit 15 shifted out from the current address register.		Xh	RW

20.2.1.7 Offset 8Bh: DMA_MPL[5-7] - DMA Memory Low Page Registers for Channels 5-7

Table 20-10. Offset 8Bh: DMA_MPL[5-7]: DMA Memory Low Page Registers for Channels 5-7

Description:	Ch. 5: 8Bh - 9Bh, Ch. 6: 89h - 99h, Ch. 7: 8Ah - 9Ah				
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 8Bh, 89h, 8Ah **Offset End:** 9Bh, 99h, 9Ah	
Size: 8 bit	**Default:** XXXXXXX			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DMALP	**DMA Low Page (ISA Address bits [23:16]):** This register works in conjunction with the DMA controller's Current Address Register to define the complete 24-bit address for the DMA channel. This register remains static throughout the DMA transfer. Bit 16 of this register is ignored when in 16 bit I/O count by words mode as it is replaced by the bit 15 shifted out from the current address register.		Xh	RW

20.2.1.8 Offset 08h: DMA_STATUS - DMA Status Register

Table 20-11. Offset 08h: DMA_STATUS - DMA Status Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 08h **Offset End:** 08h	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 18h **Offset End:** 18h	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** D0h **Offset End:** D0h	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** D1h **Offset End:** D1h	
Size: 8 bit	**Default:** XXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	CRS	**Channel Request Status:** When a valid DMA request is pending for a channel, the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware or a software request. ***Note:*** Channel 4 is the cascade channel, so the request status of channel 4 is a logical OR of the request status for channels 0 through 3. 4 Channel 0 5 Channel 1 (5) 6 Channel 2 (6) 7 Channel 3 (7)		XXXX	RO
03 : 00	CTCS	**Channel Terminal Count Status:** When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Channel 4 is programmed for cascade, so the TC bit response for channel 4 is irrelevant: 0 Channel 0 1 Channel 1 (5) 2 Channel 2 (6) 3 Channel 3 (7)		XXXX	RO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.9 Offset 0Ah: DMA_WSM - DMA Write Single Mask Register

Table 20-12. Offset 0Ah: DMA_WSM - DMA Write Single Mask Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Ah **Offset End:** 0Ah	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Ah **Offset End:** 1Ah	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** D4h **Offset End:** D4h	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** D5h **Offset End:** D5h	
Size: 8 bit	**Default:** 000001xxb			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved. Must be 0.		00000b	
02	CMS	**Channel Mask Select:** 0 = DREQ is enabled for the selected channel 1 = DREQ is disabled for the selected channel The channel is selected through bits [1:0]. Therefore, only one channel can be masked / unmasked at a time.		1	WO
01 : 00	DMACS	**DMA Channel Select:** These bits select which DMA Channel Mode Register is programmed. 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)		XX	WO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.10 Offset 0Bh: DMA_CHM - DMA Channel Mode Register

Table 20-13. Offset 0Bh: DMA_CHM - DMA Channel Mode Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Bh **Offset End:** 0Bh	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Bh **Offset End:** 1Bh	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** D6h **Offset End:** D6h	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** D7h **Offset End:** D7h	
Size: 8 bit	**Default:** 000000XXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	DMATM	**DMA Transfer Mode:** Each DMA channel can be programmed in one of four different modes: 00 Demand mode 01 Single mode 10 Reserved 11 Cascade mode		00h	WO
05	ADDIDS	**Address Increment/Decrement Select:** This bit controls address increment/decrement during DMA transfers. 0 = Address increment is selected. 1 = Address decrement is selected. Address increment is the default after part reset or Master Clear.		0h	WO
04	AUTOEN	**Autoinitialize Enable:** 0 = The autoinitialize feature is disabled and DMA transfers terminate on a terminal count. A part reset or Master Clear disables autoinitialization. 1 = The DMA restores the Base Address and Count registers to the current registers following a terminal count (TC).		0h	WO
03 : 02	DMATT	**DMA Transfer Type:** These bits represent the direction of the DMA transfer. When the channel is programmed for cascade mode, (bits[07:06] = "11") the transfer type is irrelevant. 00 Verify - No I/O or memory strobes are generated 01 Write - Data transfers from the I/O devices to memory 10 Read - Data transfers from memory to the I/O device 11 Illegal		00h	WO
01 : 00	DMACSEL	**DMA Channel Select:** These bits select the DMA Channel Mode Register that are written by bits [07:02]. 00 Channel 0 (4) 01 Channel 1 (5) 10 Channel 2 (6) 11 Channel 3 (7)		XX	WO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.11 Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register

Table 20-14. Offset 0Ch: DMA_CBP - DMA Clear Byte Pointer Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Ch **Offset End:** 0Ch	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Ch **Offset End:** 1Ch	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** D8h **Offset End:** D8h	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** D9h **Offset End:** D9h	
Size: 8 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CBP	**Clear Byte Pointer:** Command is enabled with a write to the I/O port address. Writing to this register initializes the byte pointer flip/flop to a known state. It clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers. The latch is also cleared by a part reset and by the Master Clear command. This command precedes the first access to a 16 bit DMA controller register. The first access to a 16-bit register accesses the least significant byte, and the second accesses the most significant byte.		X	WO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.12 Offset 0Dh: DMA_MC - DMA Master Clear Register

Table 20-15. Offset 0Dh: DMA_MC - DMA Master Clear Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Dh **Offset End:** 0Dh	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Dh **Offset End:** 1Dh	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** DAh **Offset End:** DAh	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** DBh **Offset End:** DBh	
Size: 8 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MSTCL	**Master Clear:** Enabled with a write to the port. This has the same effect as the hardware Reset; Command, Status, Request, and Byte Pointer flip/flop registers are cleared and the Mask Register is set.		X	WO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.13 Offset 0Eh: DMA_CM - DMA Clear Mask Register

Table 20-16. Offset 0Eh: DMA_CM - DMA Clear Mask Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Eh **Offset End:** 0Eh	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Eh **Offset End:** 1Eh	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** DCh **Offset End:** DCh	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CLMR	**Clear Mask Register:** Command enabled with a write to the port.		X	WO

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.2.1.14 Offset 0Fh: DMA_WAM - DMA Write All Mask Register

Table 20-17. Offset 0Fh: DMA_WAM - DMA Write All Mask Register

Description:					
View: IA F 1[a]	**Base Address:** 0000h (IO)			**Offset Start:** 0Fh **Offset End:** 0Fh	
View: IA F 1	**Base Address:** 0000h (IO)			**Offset Start:** 1Fh **Offset End:** 1Fh	
View: IA F 2[b]	**Base Address:** 0000h (IO)			**Offset Start:** DEh **Offset End:** DEh	
View: IA F 2	**Base Address:** 0000h (IO)			**Offset Start:** DFh **Offset End:** DFh	
Size: 8 bit	**Default:** 00001111b			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved. Must be 0.		0h	RO
03 : 00	CMSKB	**Channel Mask Bits:** Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits [03:00] are set to 1 upon part reset or Master Clear. When read, bits [03:00] indicate the DMA channel [03:00] ([07:04]) mask status. 0 Channel 0 (4) 1 Channel 1 (5) 2 Channel 2 (6) 3 Channel 3 (7) This register permits all four channels to be simultaneously enabled/disabled instead of enabling/ disabling each channel individually, as is the case with the Mask Register - Write Single Mask Bit. This register also has a read path to allow the status of the channel mask bits to be read. A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is in auto-initialization mode). Disabling channel 4 also disables channels 0–3 due to the cascade of channels 0–3 through channel 4.		1111b	RW

a. View 1 describes the control registers for Channels 0-3.
b. View 2 describes the control registers for Channels 4-7.

20.3 DMA Channel Arbitration

The IICH DMA controller consists of two logical channel groups; channels 0–3 and channels 4–7. Each group may be in either fixed or rotate mode as described in detail below.

The mode of operation for each controller is determined by the DMA Command register; address 08h for channels 0–3 and address D0h for channels 4–7. Since channels 0–3 are cascaded onto channel 4, any request on channel 0–3 appears as a request on channel 4.

The DMA controller stops rotating when an NMI is pending.

In fixed mode, the lowest numbered channel in a channel group receives highest priority. Therefore, channel 0 is the highest priority device of channels 0–3, and channel 4 is the highest priority device of channels 4–7. When both channels are programmed in fixed mode, channel 0 has highest priority and channel 7 the lowest.

In rotating mode, the lowest numbered channel starts out with highest priority. When it is serviced, the next numbered channel receives highest priority and the previous channel receives lowest priority. For example, if channel 0 has highest priority and is requesting, it wins arbitration, then is the lowest priority channel until channels 1, 2, and 3 have been serviced.

Due to the nature of channel 0–3 being cascaded onto channel 4, rotating mode adds some peculiarities to the arbitration scheme. Table 20-18 lists arbitration winners assuming all channels ares requesting.

Table 20-18. DMA Channel Priority

Current	Both Fixed	Lower Fixed, Upper Rotating	Lower Rotating, Upper Fixed	Both Rotating
0	0, 1, 2, 3, 5, 6, 7	5, 6, 7, 0, 1, 2, 3	1, 2, 3, 0, 5, 6, 7	5, 6, 7, 1, 2, 3, 0
1	0, 1, 2, 3, 5, 6, 7	5, 6, 7, 0, 1, 2, 3	2, 3, 0, 1, 5, 6, 7	5, 6, 7, 2, 3, 0, 1
2	0, 1, 2, 3, 5, 6, 7	5, 6, 7, 0, 1, 2, 3	3, 0, 1, 2, 5, 6, 7	5, 6, 7, 3, 0, 1, 2
3	0, 1, 2, 3, 5, 6, 7	5, 6, 7, 0, 1, 2, 3	0, 1, 2, 3, 5, 6, 7	5, 6, 7, 0, 1, 2, 3
5	0, 1, 2, 3, 5, 6, 7	6, 7, 0, 1, 2, 3, 5	0, 1, 2, 3, 5, 6, 7	6, 7, 0, 1, 2, 3, 5
6	0, 1, 2, 3, 5, 6, 7	7, 0, 1, 2, 3, 5, 6	0, 1, 2, 3, 5, 6, 7	7, 0, 1, 2, 3, 5, 6
7	0, 1, 2, 3, 5, 6, 7	0, 1, 2, 3, 5, 6, 7	0, 1, 2, 3, 5, 6, 7	0, 1, 2, 3, 5, 6, 7

20.4 Special Cases in Address/Count

20.4.1 Address Overrun/Underrun

Whenever the DMA is operating, the addresses do not increment or decrement through the High Page and Low Page registers. Therefore, if a 24-bit address is 01FFFFh and increments, the next address is 010000h, not 020000h. Similarly, if a 24-bit address is 020000h and decrements, the next address is 02FFFFh, not 01FFFFh.

However, when the DMA is operating in 16-bit mode, the addresses do not increment or decrement through the High Page and Low Page registers but the page boundary is now 128 Kbyte. Therefore, if a 24-bit address is 01FFFEh and increments, the next address is 000000h, not 010000h. Similarly, if a 24-bit address is 020000h and decrements, the next address is 03FFFEh, not 02FFFEh.

20.4.2 16-Bit Channels

For 16-bit channels, the DMA controller addressing is different than for 8-bit channels. The DMA controller shifts the lower 16 bits of address left 1 bit and shifts in a '0', as shown in Table 20-19. The count register is also redefined to represent words instead of bytes.

Table 20-19. Address Shifting in 16-bit DMA Transfers

Register			Address on 8 bit channels (hex)	Address on 16 bit channels (hex)
Page	High Byte	Low Byte		
00	01	01	00.01.01	00.02.02
01	FE	85	01.FE.85	01.FD.0A
01	FF	FF	01.FF.FF	01.FF.FE
00	FE	85	00.FE.85	01.FD.0A
00	FF	FF	00.FF.FF	01.FF.FE

20.4.3 Autoinitialize

By programming a bit in the DMA Channel Mode Register, a channel may be set up as an autoinitialize channel. When a channel undergoes autoinitialization, the original values of the Current Page, Current Address and Current Byte/Word Count Registers are automatically restored from the Base Page, Address, and Byte/Word Count Registers of that channel following TC. The Base Registers are loaded simultaneously with the Current Registers by the microprocessor when the DMA channel is programmed and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in autoinitialize. Following autoinitialize, the channel is ready to perform another DMA service, without processor intervention, as soon as a valid DREQ is detected.

20.4.4 Software Commands

There are three additional special software commands that the DMA controller can execute. The three software commands are:

- Clear Byte Pointer Flip-Flop
- Master Clear
- Clear Mask Register

They are independent of any specific bit pattern on the data bus.

20.5 Theory of Operation for LPC DMA

DMA on LPC is handled through the use of the LDRQ# lines from peripherals and special encodings on LAD[3:0] from the host. Single, Demand, Verify, and Increment modes are supported on the LPC interface. Channel 0–3 are 8 bit channels. Channel 5–7 are 16 bit channels. Channel 4 is reserved as a generic bus master request (see Section 20.5.1 on LPC bus masters).

20.5.1 Asserting DMA Requests

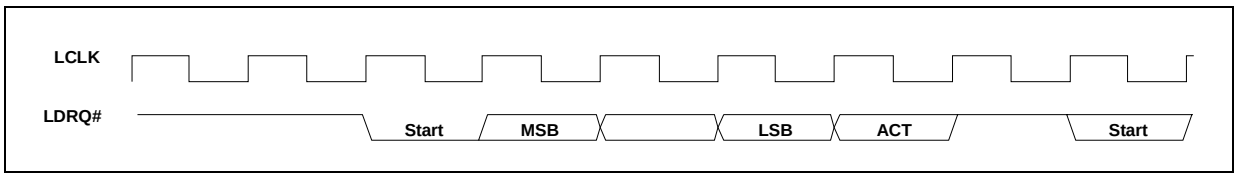
Peripherals that need DMA service encode their requested channel number on the LDRQ# signal. To simplify the protocol, each peripheral on the LPC Interface has its own dedicated LDRQ# signal (they may not be shared between two separate peripherals). The IICH has two LDRQ# inputs, allowing at least two devices to support DMA or bus mastering.

LDRQ# is synchronous with LCLK (PCI clock). As shown in Figure 20-2, the peripheral uses the following serial encoding sequence:

- Peripheral starts the sequence by asserting LDRQ# low (start bit). LDRQ# is high during idle conditions.
- The next three bits contain the encoded DMA channel number (MSB first).
- The next bit (ACT) indicates whether the request for the indicated DMA channel is active or inactive. The ACT bit is a 1 (high) to indicate if it is active and a 0 (low) if it is inactive. The case where ACT is low is rare, and is only used to indicate that a previous request for that channel is being abandoned. See Section 20.5.2 for reasons for abandoning DMA requests.
- After the active/inactive indication, the LDRQ# signal must go high for at least one clock. After that one clock, LDRQ# signal can be brought low to the next encoding sequence.

If another DMA channel also needs to request a transfer, another sequence can be sent on LDRQ#. For example, if an encoded request is sent for channel 2, and then channel 3 needs a transfer before the cycle for channel 2 is run on the interface, the peripheral can send the encoded request for channel 3. This allows multiple DMA agents behind an I/O device to request use of the LPC interface, and the I/O device does not need to self-arbitrate before sending the message.

Figure 20-2. DMA Request Assertion through LDRQ#



20.5.2 Abandoning DMA Requests

DMA Requests can be deasserted in two fashions: on error conditions by sending an LDRQ# message with the ‘ACT’ bit set to ‘0’, or normally through a SYNC field during the DMA transfer. This section describes boundary conditions where the DMA request needs to be removed prior to a data transfer. See Section 20.5.6 to see how DMA requests are terminated through a DMA transfer.

There may be some special cases where the peripheral desires to abandon a DMA transfer. The most likely case of this occurring is due to a floppy disk controller which has overrun or enduring its FIFO, or software stopping a device prematurely.

In these cases, the peripheral wishes to stop further DMA activity. It may do so by sending an LDRQ# message with the ACT bit as '0'. However, since the DMA request was seen, there is no guarantee that the cycle hasn't been granted and runs on LPC. Therefore, peripherals must take into account that a DMA cycle may still occur. The peripheral can choose not to respond to this cycle, in which case the host aborts it, or it can choose to complete the cycle normally with any random data.

This method of DMA deassertion must be prevented whenever possible in order to limit boundary conditions both on the IICH and the peripheral.

The LDREQ DMA abort scheme should not be used if a transfer is in progress (a cycle has started) and more than one transfer has been completed. In these cases, the peripheral must use the SYNC field encoding 0000.

20.5.3 General Flow of DMA Transfers

Arbitration for DMA channels is performed through the 8237 within the host. Once the host has won arbitration on behalf of a DMA channel assigned to LPC, it asserts LFRAME# on the LPC Interface and begins the DMA transfer. The general flow for a basic DMA transfer is as follows:

1. The IICH starts the transfer by asserting '0000b' on LAD[3:0] with LFRAME# asserted.
2. The IICH asserts 'cycle type' of DMA. The direction is based on the DMA transfer direction.
3. The IICH asserts the channel number and, if applicable, terminal count.
4. The IICH indicates the size of the transfer; 8 or 16 bits.
5. If a DMA reads:
   a. The IICH drives the first 8 bits of data and turns the bus around.
   b. The peripheral acknowledges the data with a valid SYNC.
   c. If a 16 bit transfer, the process is repeated for the next 8 bits.
6. If a DMA writes:
   a. The IICH turns the bus around and waits for data.
   b. The peripheral indicates data ready through SYNC and transfers the first byte.
   c. If a 16-bit transfer, the peripheral indicates data ready and transfers the next byte.
7. The peripheral turns around the bus.

20.5.4 Terminal Count

Terminal count is communicated through LAD[03] on the same clock that DMA channel is communicated on LAD[02:00]. This field is the CHANNEL field. Terminal count indicates the last byte of transfer, based upon the size of the transfer.

For example, on an 8-bit transfer size (SIZE field is '00b'), if the TC bit is set, then this is the last byte. On a 16-bit transfer (SIZE field is '01b'), if the TC bit is set, then the second byte is the last byte. The peripheral, therefore, must internalize the TC bit when the CHANNEL field is communicated and only signal TC when the last byte of that transfer size has been transferred.

20.5.5 Verify Mode

Verify mode is supported on the LPC interface. A verify transfer to the peripheral is similar to a DMA write, where the peripheral is transferring data to main memory. The indication from the host is the same as a DMA write, so the peripheral is driving data onto the LPC interface. However, the host does not transfer this data into main memory.

20.5.6 DMA Request Deassertion

An end of transfer is communicated to the IICH through a special SYNC field transmitted by the peripheral. If a DMA transfer is several bytes, such as a transfer from a demand mode device, the IICH needs to know when to deassert the DMA request based on the data currently being transferred.

The DMA agent uses a SYNC encoding on each byte of data being transferred, which indicates to the IICH whether this is the last byte of transfer or if more bytes are requested. To indicate the last byte of transfer, the peripheral uses a SYNC value of '0000b' (ready with no error), or '1010b' (ready with error). These encodings tell the IICH that this is the last piece of data transferred on a DMA read (IICH to peripheral), or the byte which follows is the last piece of data transferred on a DMA write (peripheral to IICH).

When the IICH sees one of these two encodings, it ends the DMA transfer after this byte and deasserts the DMA request to the 8237. Therefore, if the IICH indicated a 16 bit transfer, the peripheral can end the transfer after one byte by indicating a SYNC value of '0000b' or '1010b'. The IICH does not attempt to transfer the second byte, and deasserts the DMA request internally. This allows the peripheral, therefore, to terminate a DMA burst.

If the peripheral indicates a '0000b' or '1010b' SYNC pattern on the last byte of the indicated size, then the IICH only deasserts the DMA request to the 8237 since it does not need to end the transfer.

If the peripheral wishes to keep the DMA request active, then it uses a SYNC value of '1001b' (ready plus more data). This tells the 8237 that more data bytes are requested after the current byte has been transferred, so the IICH keeps the DMA request active to the 8237. Therefore, on an 8 bit transfer size, if the peripheral indicates a SYNC value of '1001b', the data is transferred and the DMA request remains active to the 8237. At a later time, the IICH starts with another START ⇒ CYCTYPE ⇒ CHANNEL ⇒ SIZE etc. combination to initiate another transfer to the peripheral.

The peripheral must not assume that the next START indication from the IICH is another grant to the peripheral if it had indicated a SYNC value of '1001b'. On a single mode DMA device, the 8237 rearbitrates after every transfer. Only demand mode DMA devices can be guaranteed that they receive the next START indication from the IICH.

Note: Indicating a '0000b' or '1010b' encoding on the SYNC field of an odd byte of a 16 bit channel (first byte of a 16 bit transfer) is an error condition.

Note: The host stops the transfer on the LPC bus as indicated, fill the upper byte with random data on DMA writes (peripheral to memory), and indicate to the 8237 that the DMA transfer occurred, incrementing the 8237s address and decrementing its byte count.

20.5.7 SYNC Field / LDRQ# Rules

Since DMA transfers on LPC are requested through an LDRQ# assertion message, and end through a SYNC field during the DMA transfer, the peripheral must obey the following rule when initiating back-to-back transfers from a DMA channel.

The peripheral must not assert another message for eight LCLKs after a deassertion is indicated through the SYNC field. This is needed to allow the 8237, which typically runs off a much slower internal clock, to see a message deasserted before it is reasserted so that it can arbitrate to the next agent.

Under default operation, the host only performs 8-bit transfers on 8-bit channels and 16-bit transfers on 16-bit channels. In order to enable 16-bit transfers on 8-bit channels, the peripheral must communicate to system BIOS that larger transfer sizes are allowed. If the host has this capability, the BIOS programs the host to attempt larger transfer sizes. The IICH does not support 32-bit DMA transfer.

The method by which this communication between host and peripheral through system BIOS is performed is beyond the scope of this specification. Since the host and peripheral are motherboard devices, no “plug-n-play” registry is required.

The peripheral must not assume that the host is able to perform transfer sizes that are larger than the size allowed for the DMA channel, and be willing to accept a SIZE field that is smaller than what it may currently have buffered.

To that end, it is recommended that future devices which may appear on the LPC bus, which require higher bandwidth than 8-bit or 16-bit DMA allow, do so with a bus mastering interface and not rely on the 8237.

§ §

21.0 Serial Peripheral Interface

21.1 Overview

The Serial Peripheral Interface (SPI) is a 4-pin interface that provides a potentially lower-cost alternative for system flash versus the Firmware Hub interface that is available on the LPC pins.

21.1.1 Features

- Support for Multiple SPI Flash Vendors
- Simple Hardware
- Equivalence to LPC-Based Firmware Hubs
  - — Provide the EP80579 Write Protection scheme
  - — Equivalent Performance (Boot time, Resume Time)
  - — Top Swap Functionality
  - — Support for E & F segments below 1 MB
  - — 64 Kb-Granular Protection

Note that the SPI does not provide support for very large BIOS sizes as easily as the FWH interface. The EP80579 SPI interface is restricted to one Chip Select pin.

The Serial Peripheral Interface (SPI) is a 4-pin interface that provides a potentially lower-cost alternative for system flash versus the Firmware Hub interface that is available on the LPC pins.

21.2 External Interface

Table 21-1. SPI Pin Interface

Signal(s)	Width	Type	IO Type	Description
SPI_SCLK	1	O	LVTTL, 3.3V	Serial bit-rate clock 17.86 MHz
SPI_CS#	1	O	LVTTL, 3.3V	CS for slave
SPI_MOSI	1	O	LVTTL, 3.3V	Master data out / Slave In
SPI_MISO	1	I	LVTTL, 3.3V	Master data in/ Slave out

Table 21-2. GPIO Boot Source Selection

GPIO17	GPIO33	Description
1	1	Default. Boot from LPC
0	1	Reserved.
1	0	Reserved.
0	0	Boot from SPI

21.3 SPI Protocol

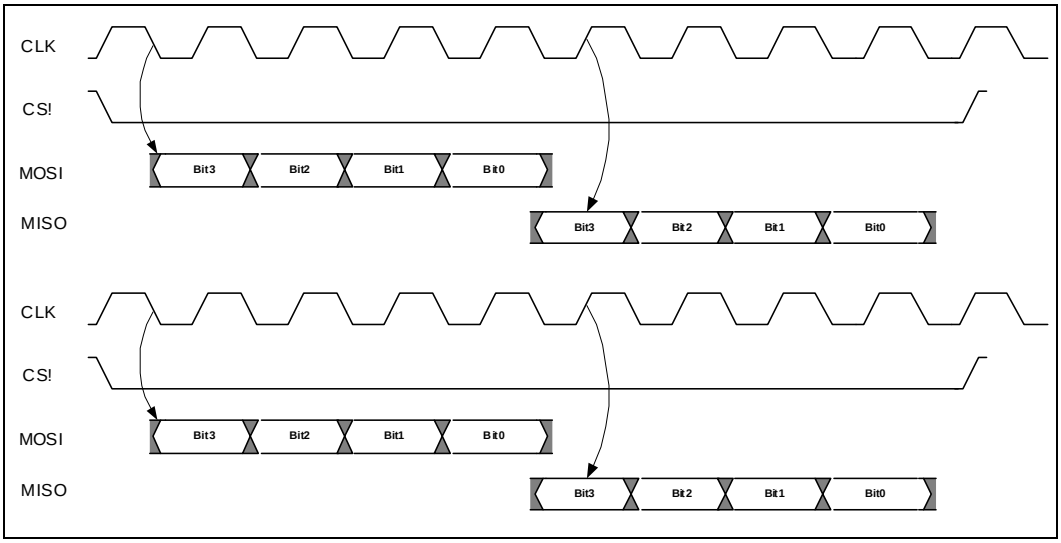
Communication on the SPI bus is done with a Master – Slave protocol. Typical bus topologies call for a single SPI Master with a single SPI Slave. The SPI interface consists of a four wire interface: clock (CLK), master data out (Master Out Slave In (MOSI)), master data in (Master In Slave Out (MISO)) and an active low chip select (CS#).

21.3.1 SPI Pin-Level Protocol

SPI communicates utilizing a synchronous protocol with the clock being driven by the Master. After selecting a Slave by asserting the SPI_CS# signal, the Master generates eight clock pulses per byte on the SPI_CLK wire, one clock pulse per data bit. [1] Data flows from master to slave on the SPI_MOSI wire and from slave to master on the SPI_MISO wire[2]. Data is setup and sampled on opposite edges of the SPI_CLK signal. Master drives data off of the falling edge of the clock and slave samples on the rising edge of the clock. Similarly, Slave drives data off of the falling edge of the clock. The master has more flexibility on sampling schemes since it controls the clock. Note that SPI_CLK flight times and the device SPI_MISO max valid times indicate that the rising edge is not feasible for sampling the SPI_MISO input at the master for a 20 MHz clock period with 50% duty cycle.

1. SPI supports 8 or 16 bit words, however all devices on the supported list only operate on 8 bit words.
2. SPI specifies that data can be shifted MSB or LSB first, however all devices on the supported list only operate MSB first.

Figure 21-1. Basic SPI Protocol



Only Mode 0 is supported.

Commands, Addresses and Data are shifted most significant bit (MSB) first. For the 24-bit address, this means bit 23 is shifted first while bit 0 is shifted last. However, for data bursts, bytes are shifted out from *least significant byte to most significant byte*, where each byte is shifted (MSB to LSB).

21.3.1.1 Addressing

A Slave is targeted for a cycle when it's SPI_CS# pin is asserted. Besides Slave addressing there is register addressing within the Slave itself. The list of EP80579 supported devices' includes only FLASH devices. See supported devices data sheets for more information.

21.3.1.2 Data Transaction

All transactions on the SPI bus must be a multiple of 8 bits. A frame consists of any number of 8-bit data packets. To initiate a data transfer, the SPI Master asserts (high to low transition) the SPI_CS# signal informing the SPI Slave that it is being targeted for a cycle. The Master will then shift out the 8-bit opcode followed by the Slave's internal address.

In the case of a Read transaction, the Slave will interpret the Slave address and begin driving data out on the SPI_MISO pin and ignore any transactions on the SPI_MOSI pin. The Master indicates Read complete by deasserting the SPI_CS# signal on an 8-bit boundary.

In the case of a Write transaction, the Slave will continue to receive Master data on the SPI_MOSI pin. The Write transaction is completed upon deassertion of the SPI_CS# signal on an 8-bit boundary.

The SPI bus does include a mechanism for flow control, however some devices include the support of a HOLD signal. See Slave documentation for more information. If the Slave receives an un-recognized or invalid opcode it should ignore the rest of the packet and wait for the deassertion of SPI_CS#.

21.3.1.3 Bus Errors

If the first 8 bits specify an opcode which is not supported the slave will not respond and wait for the next high to low transition on SPI_CS#.

SPI hardware should automatically discard 8-bit words that were not completely received upon deassertion of the SPI_CS# signal.

Any other error correction or detection mechanisms must be implemented in firmware/ software.

21.3.1.4 Instructions

Instruction	ST M25P80 (8 Mb)	ST M45PE80 (8 Mb)	NexFlash NX25P*	SST 25V040 (4Mb), SST 25VF080 (8 Mb)	Ching is (1 Mb)
Write Status	01	-	01	01	01
Data Program	02	02	02	02	02
Read Data	03	03	03	03	03
Write Disable	04	04	04	04	04
Read Status	05	05	05	05	05
Write Enable	06	06	06	06	06
Page Write	-	0A	-	-	-
Fast Read(1)	0B	0B	0B	-	0B
Ena Write Status	-	-	-	50	-
256B Erase	-	DB	-	-	-
4KByte Erase	-	-	-	20	D7
64KB Erase	D8	D8	D8	52	D8
Chip Erase	C7	-	C7	60	C7
Auto Add Inc(2)	-	-	-	AF	-
Power Down/Up	B9 / AB	B9 / AB	B9	-	-
Read ID	-	9F	90	AB or 90	AB

Notes:
1. Fast Read Protocol is not supported
2. The Auto Address Increment type is not supported.

21.3.1.5 SPI Timings

The SPI interface is designed to fall within the following protocol timing specs. These specs are intended to operate with most SPI Flash devices.

Table 21-3. SPI Cycle Timings

Parameter	Minimum Value	Description
SPI_CS# Setup	30 ns	SPI_CS# low to SPI_CLK high
SPI_CS# Hold	30 ns	SPI_CLK low to SPI_CS# low
Clock High	22 ns	Time that SPI_CLK is Driven high per clock period
Clock Low	22 ns	Time that SPI_CLK is Driven Low per clock period

21.4 Host Side Interface

21.4.1 SPI Host Interface Registers

The SPI Host Interface registers are memory-mapped in the RCRB Chipset Memory Space in the range 3020h to 308Fh.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

The table below does NOT include the 3020h offset.

21.4.2 Register Overview

Table 21-4. Bus 0, Device 31, Function 0, PCI Registers Mapped Through RCBA Bar

Offset Start	Offset End	Register ID - Description	Default Value
3020h	3021h	"Offset 3020h: SPIS - SPI Status" on page 790	0001h
3022h	3023h	"Offset 3022h: SPIC - SPI Control" on page 791	2005h
3024h	3027h	"Offset 3024h: SPIA - SPI Address" on page 792	00XXXXXh
3028h	302Bh	"Offset 3028h: SPID0 - SPI Data 0" on page 792	XXXXXXXXh
3030h at 4h	306Ch at 4h	"Offset 3030h, 3038h, 3040h, 3048h, 3050h, 3058h, 3060h: SPI[0-6] - SPI Data [0-6]" on page 793	00000000h
3070h	3073h	"Offset 3070h: BBAR - BIOS Base Address" on page 793	00000000h
3074h	3075h	"Offset 3074h: PREOP - Prefix Opcode Configuration" on page 794	0004h
3076h	3077h	"Offset 3076h: OPTYPE - Op Code Type" on page 794	0000h
3078h	307Fh	"Offset 3078h: OPMENU - OPCODE Menu Configuration" on page 795	00000005h
3080h at 4h	3083h at 4h	"Offset 3080h: PBR0 - Protected BIOS Range #0" on page 796	00000000h

21.4.2.1 Offset 3020h: SPIS – SPI Status

Table 21-5. Offset 3020h: SPIS - SPI Status

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3020h **Offset End:** 3021h	
Size: 16 bit	**Default:**	0001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	SCL	**SPI Configuration Lock-Down:** When set to 1, the SPI Static Configuration information in offsets 50h through 6Bh can not be overwritten. Once set to 1, this bit can only be cleared by a hardware reset.		0	RWL
14 :4	Rsvd	Reserved		0	RV
3	BAS	**Blocked Access Status:** Hardware sets this bit to 1 when an access is blocked from running on the SPI interface due to one of the protection policies or when any of the programmed cycle registers are written while a programmed access is already in progress. This bit is set for both programmed accesses and direct memory reads that get blocked. This bit remains asserted until cleared by software writing a 1 or hardware reset.		0	RWC
2	CDS	**Cycle Done Status:** The EP80579 sets this bit to 1 when the SPI Cycle completes (i.e., SCIP bit is 0) after software sets the GO bit. This bit remains asserted until cleared by software writing a 1 or hardware reset. When this bit is set and the SPI bit in Offset 3022h: SPIC – SPI Control is set, an internal signal is asserted to the SMI# generation block. Software must make sure this bit is cleared prior to enabling the SPI SMI# assertion for a new programmed access. This bit gets set after the Status Register Polling sequence completes after reset deasserts. It is cleared before and during that sequence.		0	RWC
1	Rsvd	**Reserved.**		0	RV
0	SCIP	**SPI Cycle In Progress (SCIP):** Hardware sets this bit when software sets the SPI Cycle Go bit in the Offset 3022h: SPIC – SPI Control. This bit remains set until the cycle completes on the SPI interface. Hardware automatically sets and clears this bit so that software can determine when read data is valid and/or when it is safe to begin programming the next command. Software must only program the next command when this bit is 0. This bit reports 1b during the Status Register Polling sequence after reset deasserts; it is cleared when that sequence completes.		1	RO

21.4.2.2 Offset 3022h: SPIC – SPI Control

Table 21-6. Offset 3022h: SPIC - SPI Control

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3022h **Offset End:** 3023h	
Size: 16 bit	**Default:**	2005h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	SSMIE	**SPI SMI# Enable:** When set to 1, the SPI asserts an SMI# request whenever the Cycle Done Status bit is 1.		00	RW
14	DC	**Data Cycle:** When set to 1, there is data that corresponds to this transaction. When 0, no data is delivered for this cycle, and the DBC and data fields themselves are don't cares.		1	RW
13 :08	DBC	**Data Byte Count (DBC):** This field specifies the number of bytes to shift in or out during the data portion of the SPI cycle. The valid settings (in decimal) are any value from 0 to 63. The number of bytes transferred is the value of this field plus 1. Note that when this field is 00_0000b, then there is 1 byte to transfer and that 11_1111b means there are 64 bytes to transfer.		0	RW
7	Rsvd	Reserved		0	RV
6 :4	COP	**Cycle Opcode Pointer:** This field selects one of the programmed opcodes in the Offset 3078h: OPMENU – Opcode Menu Configuration to be used as the SPI Command/Opcode. In the case of an Atomic Cycle Sequence, this determines the second command.		0	RW
3	SPOP	**Sequence Prefix Opcode Pointer:** This field selects one of the two programmed prefix opcodes for use when performing an Atomic Cycle Sequence. A value of 0 points to the opcode in the least significant byte of the Offset 3074h: PREOP – Prefix Opcode Configuration register. By making this programmable, the EP80579 supports flash devices that have different opcodes for enabling writes to the data space vs. status register.		0	RW
2	ACS	**Atomic Cycle Sequence (ACS):** When set to 1 along with the SCGO assertion, the EP80579 will execute a sequence of commands on the SPI interface. The sequence is composed of: Atomic Sequence Prefix Command (8-bit opcode only) Primary Command specified by software (can include address and data) Polling the Flash Status Register (opcode 05h) until bit 0 becomes 0b. The SPI Cycle in Progress bit remains set and the Cycle Done Status bit in Offset 3020h: SPIS – SPI Status register remains unset until the Busy bit in the Flash Status Register returns 0.		1	RW
1	SCGO	**SPI Cycle Go (SCGO):** This bit always returns 0 on reads. However, a write to this register with a '1' in this bit starts the SPI cycle defined by the other bits of this register. The SPI Cycle in Progress (SCIP) bit in Offset 3020h: SPIS – SPI Status register gets set by this action. Hardware must ignore writes to this bit while the SPI Cycle In Progress bit is set. Hardware allows other bits in this register to be programmed for the same transaction when writing this bit to 1. This saves an additional memory write.		0	RWS
0	Rsvd	**Reserved**		1	RV

21.4.2.3 Offset 3024h: SPIA – SPI Address

Table 21-7. Offset 3024h: SPIA - SPI Address

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3024h **Offset End:** 3027h	
Size: 32 bit	**Default:**	00XXXXXh		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RSVD	Reserved		0	RV
23 :00	SCA	**SPI Cycle Address (SCA):** This field is shifted out as the SPI Address (MSB first).		0	RW

21.4.2.4 Offset 3028h: SPID0 – SPI Data 0

Table 21-8. Offset 3028h: SPID0 - SPI Data 0

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3028h **Offset End:** 302Bh	
Size: 64 bit	**Default:**	XXXXXXXXh		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :00	SCD	**SPI Cycle Data 0 (SCD0):** This field is shifted out as the SPI Data on the Master-Out Slave-In Data pin (SPI_MOSI) during the data portion of the SPI cycle. This register also shifts in the data from the Master-In Slave-Out pin (SPI_MISO) into this register during the data portion of the SPI cycle. The data is always shifted starting with the least significant byte, MSB to LSB, followed by the next least significant byte, MSB to LSB, etc. Specifically, the shift order on SPI in terms of bits within this register is: 7-6-5-4-3-2-1-0-15-14-13-...8-23-22-...16-31...24-39..32...etc. Bit 56 is the last bit shifted out/in. There are no alignment assumptions; byte 0 always represents the value specified by the cycle address. Note that the data in this register may be modified by the hardware during any programmed SPI transaction. Direct Memory Reads do not modify the contents of this register. (This last requirement is needed in order to properly handle the collision case described in Section 21.4.3.2.) This register is initialized to 0 by the reset assertion. However, the least significant byte of this register is loaded with the first Status Register read of the Atomic Cycle Sequence that the hardware automatically runs out of reset. Therefore, bit 0 of this register can be read later to determine if the platform encountered the boundary case in which the SPI flash was busy with an internal instruction when the platform reset deasserted.		see description	RW0

21.4.2.5 SPID[0-6] – SPI Data N

Table 21-9. Offset 3030h, 3038h, 3040h, 3048h, 3050h, 3058h, 3060h: SPI[0-6] - SPI Data [0-6]

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3030h at 4h **Offset End:** 306Ch at 4h	
Size: 64 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 : 00	SCD	**SPI Cycle Data N (SCD[N]):** Similar definition as SPI Cycle Data 0. However, this register does not begin shifting until SPID[N-1] has completely shifted in/out.		0	RW

21.4.2.6 Offset 3070h: BBAR – BIOS Base Address

This register is not writable when the SPI Configuration Lock-Down bit in Offset 3020h: SPIS – SPI Status register is set.

Table 21-10. Offset 3070h: BBAR - BIOS Base Address

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3070h **Offset End:** 3073h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Rsvd	Reserved		0	RV
23 :08	BSP	**Bottom of System Flash:** This field determines the bottom of the System BIOS. The EP80579 will not run programmed commands nor memory reads whose address field is less than this value. This field corresponds to bits 23:8 of the 3-byte address; bits 7:0 are assumed to be 00h for this vector when comparing to a potential SPI address. Software must always program 1's into the upper, Don't Care bits of this field based on the flash size. Hardware does not know the size of the flash array and relies upon the correct programming by software. The default value of 0000h results in all cycles allowed. Note: The SPI Host Controller prevents any Programmed cycle using the Address Register with an address less than the value in this register. Some flash devices specify that the Read ID command must have an address of 0000h or 0001h. If this command must be supported with these devices, it must be performed with the BBAR - BIOS Base Address programmed to 0h. Some of these devices have actually been observed to ignore the upper address bits of the Read ID command.		0	RWS
7 :00	Rsvd	Reserved		0	RV

21.4.2.7 Offset 3074h: PREOP – Prefix Opcode Configuration

This register is not writable when the SPI Configuration Lock-Down bit in Offset 3020h: SPIS – SPI Status register is set.

Table 21-11. Offset 3074h: PREOP - Prefix Opcode Configuration

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3074h **Offset End:** 3075h	
Size: 16 bit	**Default:**	0004h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	PO1	**Prefix Opcode 1:** Software programs an SPI opcode into this field that is permitted to run as the first command in an atomic cycle sequence.		0	RWS
7 :00	PO0	**Prefix Opcode 0:** Software programs an SPI opcode into this field that is permitted to run as the first command in an atomic cycle sequence.		04h	RWS

21.4.2.8 Offset 3076h: OPTYPE – Opcode Type Configuration

This register is not writable when the SPI Configuration Lock-Down bit in Offset 3020h: SPIS – SPI Status register is set. Entries in this register correspond to the entries in the Offset 3078h: OPMENU – Opcode Menu Configuration register. Note that the definition below only provides write protection for opcodes that have addresses associated with them. Therefore, any erase or write opcodes that do not use an address should be avoided (for example, “Chip Erase” and “Auto-Address Increment Byte Program”).

Table 21-12. Offset 3076h: OPTYPE - Op Code Type (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3076h **Offset End:** 3077h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :14	OT7	**Opcode Type 7:** See the description for bits 1:0		0	RWS
13 :12	OT6	**Opcode Type 6:** See the description for bits 1:0		0	RWS
11 :10	OT5	**Opcode Type 5:** See the description for bits 1:0		0	RWS
9 :08	OT4	**Opcode Type 4:** See the description for bits 1:0		0	RWS
7 :06	OT3	**Opcode Type 3:** See the description for bits 1:0		0	RWS

Table 21-12. Offset 3076h: OPTYPE - Op Code Type (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3076h **Offset End:** 3077h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
5 :04	OT2	**Opcode Type 2:** See the description for bits 1:0		0	RWS
3 :02	OT1	**Opcode Type 1:** See the description for bits 1:0		0	RWS
1 :00	OT0	**Opcode Type 0:** This field specifies information about the corresponding Opcode 0. This information allows the hardware to 1) know whether to use the address field and 2) provide BIOS protection capabilities. The hardware implementation also uses the read vs. write information for modifying the behavior of the SPI interface logic. The encoding of the two bits is: 00 = No Address associated with this Opcode and Read Cycle type 01 = No Address associated with this Opcode and Write Cycle type 10 = Address required; Read cycle type 11 = Address required; Write cycle type		0	RWS

21.4.2.9 Offset 3078h: OPMENU – Opcode Menu Configuration

This register is not writable when the SPI Configuration Lock-Down bit in Offset 3020h: SPIS – SPI Status register is set. Eight entries are available in this register to give BIOS a sufficient set of commands for communicating with the flash device, while also restricting what malicious software can do. This keeps the hardware flexible enough to operate with a wide variety of SPI devices.

It is recommended that BIOS avoid programming Write Enable opcodes in this menu. Malicious software could then perform writes and erases to the SPI flash without using the atomic cycle mechanism. Write Enable opcodes should only be programmed in the Offset 3074h: PREOP – Prefix Opcode Configuration.

Table 21-13. Offset 3078h: OPMENU - OPCODE Menu Configuration (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3078h **Offset End:** 307Fh	
Size: 64 bit	**Default:** 00000005h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :56	AO7	**Allowable Opcode 7:** See the description for bits 7:0		0	RWS
55 :48	AO6	**Allowable Opcode 6:** See the description for bits 7:0		0	RWS
470 :40	AO5	**Allowable Opcode 5:** See the description for bits 7:0		0	RWS
39 :32	AO4	**Allowable Opcode 4:** See the description for bits 7:0		0	RWS
31 :24	AO3	**Allowable Opcode 3:** See the description for bits 7:0		0	RWS

Table 21-13. Offset 3078h: OPMENU - OPCODE Menu Configuration (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3078h **Offset End:** 307Fh	
Size: 64 bit	**Default:**	00000005h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 :16	AO2	**Allowable Opcode 2:** See the description for bits 7:0		0	RWS
15 :08	AO1	**Allowable Opcode 1:** See the description for bits 7:0		0	RWS
7 :00	AO0	**Allowable Opcode 0:** Software programs an SPI opcode into this field for use when initiating SPI commands through the Control Register.		05h	RWS

21.4.2.10 Offset 3080h: PBR0 – Protected BIOS Range [0-2]

This register can not be written when the SPI Configuration Lock-Down bit in Offset 3020h: SPIS – SPI Status register is set to 1.

Table 21-14. Offset 3080h: PBR0 - Protected BIOS Range #0

Description:					
View: PCI	**BAR:** RCBA		**Bus:Device:Function:** 0:31:0	**Offset Start:** 3080h at 4h **Offset End:** 3083h at 4h	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	WPE	**Write Protection Enable:** When set, this bit indicates that the Base and Limit fields in this register are valid and that writes directed to addresses between them (inclusive) must be blocked by hardware. The base and limit fields are ignored when this bit is cleared.		0	RW-Special
30 :24	Reserved	Reserved		0	RV
23 :12	PRL	**Protected Range Limit:** This field corresponds to SPI address bits 23:12 and specifies the upper limit of the protected range. Address bits 11:0 are assumed to be FFFh for the limit comparison. Any address greater than the value programmed in this field is unaffected by this protected range.		000h	RW- Special
11 :00	PRB	**Protected Range Base:** This field corresponds to SPI address bits 23:12 and specifies the lower base of the protected range. Address bits 11:0 are assumed to be 000h for the base comparison. Any address less than the value programmed in this field is unaffected by this protected range.		000h	RW- Special

21.4.3 Running SPI Cycles from the Host

21.4.3.1 Memory Reads

Memory Reads to the BIOS Range result in a READ command (03h) with the lower 3 bytes of the address delivered in the SPI cycle. By sending the entire 24 bits of address out to the SPI interface unchanged, the EP80579 hardware can support various flash memory sizes without having straps or automatic detection algorithms in hardware. The flash memory device must ignore the upper address bits such that an address of FFFFFFh simply aliases to the top of the flash memory. This is true for all supported flash devices. When considering additional flash parts, this behavior should be checked.

For compatibility with the FWH interface, the SPI interface supports decoding the two 64 KB BIOS ranges at the E0000h and F0000h segments just below 1 MB. These ranges must be re-directed (aliased) to the ranges just below 4 GB by the EP80579. This is done by forcing the upper address bits (23:20) to 1s when performing the read on the SPI interface.

When the SPI Prefetch Enable bit in Offset DCh: BC: BIOS Control Register is set, the EP80579 checks if the starting address for a read is aligned to the start of a 64B block (i.e., address bits 5:0 are 00h). If this is not the case, then the EP80579 only reads the length specified by the current read. If the read is aligned to the start of the 64B block with the SPI Prefetch Enable bit set, then the read burst continues on the SPI pins until 64 bytes have been received. Note that the EP80579 always performs the entire 64B burst when the conditions are met to perform the prefetch when the memory read request is received. This policy can result in a large penalty if the read addresses are not sequential. Software is allowed and encouraged to dynamically turn on prefetching only when the reads are sequential (for example, if shadowing the BIOS using consecutive DWord reads).

When prefetching is enabled, the read buffer must be enabled for caching. If the EP80579 detects a read to the range that is currently in (or being fetched for) the read buffer, it will not perform another read cycle on the SPI pins. Instead, the data is returned from the read buffer. Note that the entire read request must be contained in the cache in order to avoid running the read on the SPI interface.

The following events invalidate the read buffer “cache”:

1. A Programmed Access begins. Note that if the cycle is blocked from running for protection or other reasons, the cache is not flushed.
2. A Memory read to a BIOS range that does not hit the range in the read buffer.
3. System Reset.
4. Software setting the Cache Disable bit (and clearing the Prefetch Enable) in Offset DCh: BC: BIOS Control Register. This can serve as a way to flush the cache in software.

Even when prefetching is disabled, the read buffer can act as a cache for Direct Memory Read Data. This is a potentially valuable boot-time optimization that leverages the basic caching mechanism that is needed for prefetching anyway. The cache is loaded with the data received on every Direct Memory Read that runs on the SPI pins. That data remains valid within the cache until any one of the conditions listed above occurs. For the cache to work properly, the Direct Memory Read must be fully contained within a 64-byte aligned range. The following events result in a valid read buffer cache when the caching is enabled:

1. A host read to the SPI BIOS with a length of 64 Bytes. This cycle must be aligned to a 64B boundary.
2. A host read to the SPI BIOS of any length with a 64B-aligned address and prefetching is enabled.

Note that, although the SPI interface may “burst ahead” for up to 64 bytes, the Host Interface may still have to wait for prefetched data to arrive from the flash before generating the completion back to the processor. The round trip delay for the platform to complete one DWord and run the host read for the next sequential DWord can be shorter than the SPI time to receive another 32 bits.

If a Direct Memory Read targeting the SPI flash is received while the host interface is already busy with either another Direct Memory Read or a Programmed Access, then the SPI Host hardware will hold the new Direct Memory Read (and the host processor) pending until the preceding SPI access completes. Note that it is possible for a second Direct Memory Read to be received while the prefetching continues for a first Direct Memory Read.

The SPI interface provides empty flash detection equivalent to FWH (i.e., all 1’s on the initial boot access.)

It is possible that a Direct Memory Read targeting the SPI flash can be issued with non-contiguous byte enables. While the CPU cannot create these cycles, peer agents can.

The SPI interface handles these Direct Memory Read transactions in the following fashion. Note that the byte enables in the table are active high, and BE[3] is the most significant byte enable of the DWord.

Table 21-15. Byte Enable Handling on Direct Memory Reads

# DWords Requested	First DWord BE[3:0]	Last DWord BE[3:0]	Action Taken
1	0000	Don’t Care	Zero bytes read from SPI, no SPI transaction started
1	0001, 0010, 0100, 1000, 0011, 0110, 1100, 0111, 1110, 1111	Don’t Care	Bytes read from SPI = bytes requested starting from lowest requested byte
1	0101, 1001, 1010, 1011, 1101	Don’t Care	Full DW (4 bytes) requested from SPI
>1	0000	Don’t Care	Undefined behavior. Illegal protocol
>1	1000, 1100, 1110, 1111	Don’t Care	Bytes read from SPI = 4* (num DW -1) + bytes requested in first DW. Address starts from lowest requested byte.
>1	0001, 0010, 0011, 0100, 0101, 0110, 0111, 1001, 1010, 1011, 1101	Don’t Care	Bytes read from SPI = 4* num DW

When coming out of a platform reset, the SPI Host Controller must hold the initial Direct Memory Read from the processor pending until the SPI flash is no longer busy with an internal write or erase instruction. In order to achieve this, the host controller reads the Status Register (opcode = 05h) of the flash device until bit 0 is cleared. This is equivalent to the polling performed following an atomic cycle, during which the Direct Memory Reads are held pending. Depending on the type of flash and type of long instruction performed, the delay could be long enough to cause a watchdog timeout in the processor or chipset. Although this error condition is deemed acceptable in response to this rare error scenario (reset during flash update), it can be avoided altogether by selecting flash instructions on SPI devices that complete in less than ~1 second. Note that in the typical boot case, the status read on the SPI interface will complete well before the processor boot fetch due to the delay from PLTRST# deassertion to CPURST# deassertion.

21.4.3.2 Generic Programmed Commands

All commands other than the standard (memory) reads must be programmed by the BIOS in the SPI Control, address, data, and opcode configuration registers in Section 21.4.1. The opcode type in Offset 3076h: OPTYPE – Opcode Type Configuration and data byte count fields in Offset 3074h: PREOP – Prefix Opcode Configuration determine how many clocks to run before deasserting the chip enable. The flash data is always shifted in for the number of bytes specified and the BIOS out data is always shifted out for the number of data bytes specified. Note that the hardware restricts the burst lengths that are allowed.

The status bit in Offset 3020h: SPIS – SPI Status indicates when the cycle has completed on the SPI port allowing the host to know when read results can be checked and/or when to initiate a new command.

The EP80579 also provides the “Atomic Cycle Sequence” for performing erases and writes to the SPI flash in Offset 3022h: SPIC – SPI Control. When this bit is 1 (and the SPI Cycle Go bit is written to 1), a sequence of cycles is performed on the SPI interface. In this case, the specified cycle is preceded by the Prefix Command (8-bit programmable opcode) and followed by repeated reads to the Status Register (opcode 05h) until bit 0 indicates the cycle has completed. The hardware does not attempt to check that the programmed cycle is a write or erase.

If a Programmed Access is initiated (SPI Cycle Go written to 1) while the SPI host interface logic is already busy with a Direct Memory Read, then the SPI Host hardware will hold the new Programmed Access pending until the preceding SPI access completes. It will then begin to request the SPI bus for the Programmed Access.

Once the SPI Host hardware has committed to running a programmed access, subsequent writes to the programmed cycle registers that occur before it has completed will not modify the original transaction and will result in the assertion of the Blocked Access Status bit in Offset 3020h: SPIS – SPI Status. Software should never purposely behave in this way and rely on this behavior. However, the Blocked Access Status bit provides basic error-reporting in this situation. Writes to the following registers cause the Blocked Access Status bit assertion in this situation:

- Offset 3022h: SPIC – SPI Control
- Offset 3024h: SPIA – SPI Address
- Offset 3028h: SPID0 – SPI Data 0

21.4.3.3 Flash Protection

There are two types of Flash Protection mechanisms:

1. BIOS Range Write Protection
2. SMI#-Based Global Write Protection

The two mechanisms are conceptually ORed together such that if any of the mechanisms indicate that the access should be blocked, then it is blocked. Table 21-16 provides a summary of the three mechanisms.

Table 21-16. Flash Protection Mechanism Summary

Mechanism	Accesses Blocked	Range Specific	Reset-Override or SMI#-Override	Equivalent Function on FWH
BIOS Range Write Protection	Writes	Yes	Reset Override	FWH Sector Protection
SMI#-Based Global Write Protection	Writes	No	SMI# Override	Same as Write Protect in previous chipsets for FWH

The EP80579 provides these protections in hardware. Note that it is critical that the hardware must not allow malicious software to modify the address or opcode pointers after determining that a cycle is allowed to run, such that the actual cycle that runs on SPI should have been blocked.

If the command associated with an atomic cycle sequence is blocked according to the EP80579 configuration, the EP80579 must not run any of the sequence.

A blocked command will appear to software to finish, except that the Blocked Access Status bit in Offset 3020h: SPIS – SPI Status register is set in this case.

21.4.3.3.1 BIOS Range Write Protection

The EP80579 provides a method for blocking writes to specific ranges in the SPI flash when the Protected BIOS Ranges are enabled. This is achieved by checking the Opcode type information (which can be locked down by the initial Boot BIOS) and the address of the requested command against the base and limit fields of a Write Protected BIOS range.

Only the initial address is checked. Since writes wrap within a page, there should be no issue with writes illegally occurring in the next page (assuming the BIOS has configured the Protection Limit to align with the edge of a page).

Note that once BIOS has locked down the Protected BIOS Range registers, this mechanism remains in place until the next system reset.

21.4.3.3.2 SMI# Based Global Write Protection

The EP80579 provides a method for blocking writes to the SPI flash when the Write Protect bit is cleared (i.e., protected) in Offset DCh: BC: BIOS Control Register. This is achieved by checking the Opcode type information (which can be locked down by the initial Boot BIOS) of the requested command.

The Write Protect and Lock Enable bits interact in the same manner for SPI BIOS as they do for the FWH BIOS.

21.4.3.4 Decoding Memory Ranges for SPI

The Boot BIOS Destination straps are sampled on the rising edge of PWROK. If the SPI port is selected, then the Section 19.2.4, “LPC I/O Configuration Registers” on page 747 to determine what ranges of memory read addresses are forwarded to SPI. (See Section 3.7 for details of the actual pin(s) used for selecting SPI.) The Feature space ranges are unique to the FWH flash. However, the feature space can be treated just like standard memory from an SPI perspective and therefore allow up to 16 MB of contiguous memory decode. The EP80579 forwards both data and feature space ranges to the SPI interface (although the BIOS BAR may block the feature space accesses in situations where the flash size is less than 4 MB). Since there is only one Flash Chip Enable pin on the EP80579, there is no need to map the various flash ranges to multiple enables. Of course, in order to utilize 16 MB, the single flash device would need to support 128 Mbits of data.

The Top Swap mechanism works in the same way that it does on LPC. Address bit 16 is inverted when Top Swap is enabled for any accesses to the upper two 64 KB blocks. Also like LPC, the Top Swap functionality does not apply to accesses generated to the holes below 1 MB. The SPI interface performs the address bit inversion on only the Direct Memory Read access method; software can control the address directly with the programmed command access method. The prefetching and caching logic consistently comprehends the address inversion to avoid delivering bad data. Also, the protection mechanisms described above observe the address after the inversion logic.

Memory writes to the BIOS memory range are dropped. This forces all of these potentially harmful cycles to go through the Programmed Commands interface.

Note that Direct Memory Reads to the E0000h-FFFFFh segments are remapped to top of flash as mentioned previously in section Section 21.4.3.1. This range is not remapped when using Programmed Accesses.

21.5 BIOS Programming Considerations

21.5.1 SPI Initialization

This section provides a high level description of the steps that the BIOS should take upon coming out of RESET when using SPI Flash.

1. Boot vector fetch and other initial BIOS reads using Direct Memory Reads (some of which are 64 byte code reads). Caching is enabled in hardware by default to improve performance on consecutive reads to the same line.
2. Turn on the SPI Prefetching policy in the LPC Bridge Configuration Space (Offset DCh: BC: BIOS Control Register). This policy bit is in configuration space to avoid requiring protected memory space early in the boot process.
3. Copy the various BIOS modules out of the SPI Flash using Direct Memory Reads. It is assumed that these reads are shorter than 64 bytes and are targeted to consecutive addresses; hence, the prefetch mechanism improves the performance of this sequence.
4. Turn off the SPI Prefetch policy.
5. Program opcode registers in order to discover which Flash device is being used. Four of the six supported Flash devices support the READ ID instruction. Details of the discovery algorithm are outside the scope of this specification.
6. Disable Future Request, Offset 3022h: SPIC – SPI Control bit 0. Default state is Future Request enabled.
7. Re-program opcode registers to support specific Flash vendor’s commands. If not using all of the Opcode Menu and Prefix Opcodes, BIOS should program a “safe”

value in the unused opcodes to minimize what malicious software can do. A suggested safe value is to replicate one of the valid entries.

a. Offset 3074h: PREOP – Prefix Opcode Configuration
b. Offset 3076h: OPTYPE – Opcode Type Configuration
c. Offset 3078h: OPMENU – Opcode Menu Configuration

8. Setup protection registers as needed.
   a. Offset 3070h: BBAR – BIOS Base Address
   b. Offset 3080h: PBR0 – Protected BIOS Range [0-2]
9. Lock down the SPI registers, Offset 3020h: SPIS – SPI Status bit 15.
10. Set Up SMI based write protection as needed (same as FWH).

§ §

22.0 General Purpose I/O: Bus 0, Device 31, Function 0

22.1 Overview

- The following GPIO pins are implemented:

Note: Pin direction is related to GPIO mode.

Input Only : 22 (00:15, 26, 29:31, 40:41)

Output Only : 8 (16:21, 23, 48)

Input/Output : 6 (24:25, 27:28, 33:34).

- GPIO pins [22, 32, 35:39, 42:47, 50:63] do not exist.
- Some of the GPIO pins can be configured to be used for alternate functions. The alternate function and IRQ mapping of function multiplexed GPIO pins is provided in Table 22-1.
- GPIO pins [16:21, 23:25, 27:28, 30:31, 33:34, 40] are configured to generate IRQ interrupt to IO-APIC if ETR3.GPIO_IRQ_STRAP_STS = 1 [strap pulling SIU2_TXD to Low on the rising edge of PWROK). But BIOS can still change the function of each of these pins to GPIO mode by programming the individual bits in GPIO_USE_SELx register.
- Bits in GPIO_USE_SELx register enable GPIO[n] (where n is the bit number) to be used as a GPIO, rather than for the alternative function. Pin is used as GPIO when the individual bit in GPIO_USE_SELx register is set, otherwise used as alternative function. Example: Software sets bit 2 in this register to enable GPIO[2] (instead of using that signal for PIRQ[E]#).
- If GPIO[n] does not exist, then the bit in GPIO_USE_SELx register is always read as 0 and writes have no effect. Example: Bit 22 in GPIO_USE_SEL1 is not supported because there is no corresponding GPIO[22].
- GPIO pins [0:1, 6:10, 12:15, 48] do not have an alternative function.
- GPIO pins can not generate IRQ interrupt when ETR3.GPIO_IRQ_STRAP_STS = 0.
- When configured to IRQ mode, the pin direction of the relevant pins are set to “input” by GPIO logic. ICH6 compatibility is expected when the GPIOs are in GPIO mode.
- When configured to GPIO mode, the muxing logic presents the inactive state to alternate mode logic that uses the pin as an input.
- After a full reset (RSMRST#), all multiplexed signals in the resume and core wells are configured as GPIO rather than their alternative function.

Table 22-1. GPIO Pin's Alternative Function

GPIO PINs	GPIO Mode Capability	Default Pin direction (GPIO Mode)	Default Level Driven Out (GPIO Mode)	Alternative Function	Interrupt Capability			Pin direction when in alternative mode
					SMI# / SCI	IRQ	Wake Event	
1 : 0	Input Only	Input	N/A	None	Yes	No	No	N/A
2	Input Only	Input	N/A	PIRQE#	Yes	Yes-PIRQ	No	Input
3	Input Only	Input	N/A	PIRQF#	Yes	Yes-PIRQ	No	Input
4	Input Only	Input	N/A	PIRQG#	Yes	Yes-PIRQ	No	Input
5	Input Only	Input	N/A	PIRQH#	Yes	Yes-PIRQ	No	Input
7 : 6	Input Only	Input	N/A	None	Yes	No	No	N/A
11 : 8	Input Only	Input	N/A	None	Yes	No	Yes	N/A
13 : 12	Input Only	Input	N/A	None	Yes	No	No	N/A
15 : 14	Input Only	Input	N/A	None	Yes	No	Yes	N/A
16	Output Only	Output	High	IRQ[24]	No	Yes	No	Input
17	Output Only	Output	High	IRQ[25]	No	Yes	No	Input
18	Output Only	Output	High	IRQ[36]	No	Yes	No	Input
19	Output Only	Output	High	IRQ[37]	No	Yes	No	Input
20	Output Only	Output	High	IRQ[26]	No	Yes	No	Input
21	Output Only	Output	High	IRQ[27]	No	Yes	No	Input
22	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A
23	Output Only	Output	Low	IRQ[28]	No	Yes	No	Input
24	I/O	Output	High	IRQ[29]	No	Yes	No	Input
25	I/O	Output	High	IRQ[38]	No	Yes	No	Input
26	Input Only	Input	N/A	SATA	No	No	No	Input
27	I/O	Output	High	IRQ[39]	No	Yes	No	Input
28	I/O	Output	High	IRQ[30]	No	Yes	No	Input
29	Input Only	Input	N/A	SATA	No	No	No	Input
30	Input Only	Input	N/A	IRQ[31]	No	Yes	No	Input
31	Input Only	Input	N/A	IRQ[32]	No	Yes	No	Input
32	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A
33	I/O	Output	High	IRQ[33]	No	Yes	No	Input
34	I/O	Output	High	IRQ[34]	No	Yes	No	Input
39 : 35	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A
40	Input Only	Input	N/A	IRQ[35]	No	Yes	No	Input
41	Input Only	Input	N/A	Idreq1	No	No	No	Input
47 : 42	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A
48	Output Only	Output	High	None	No	No	No	N/A
49	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A
63 : 50	Not Implemented	N/A	N/A	N/A	N/A	N/A	N/A	N/A

22.1.1 GPIO Summary Table

Some GPIOs exist in the resume power plane. Care must be taken to make sure GPIO signals are not driven high into powered-down planes.

Some GPIOs may be connected to pins on devices that exist in the core well. If these GPIOs are outputs, there is a danger that a loss of core power (PWROK low) or a Power Button Override event results inCMI driving a pin to a logic '1' to another device that is powered down.

The following table summarizes the GPIOs.

Table 22-2. GPIO Summary Table (Sheet 1 of 2)

GPIO#	PowerWell	Muxed	GPIO_USE_SEL	GP_IO_SEL	GP_LVL	GPO_BLINK	GPI_INV
0	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
1	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
2:5	Core	Yes	1 when GPIO, 0 when PIRQ[E-H]#	Always 1	Always 0	Always 0	RW – D=0
6	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
7	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
8	Resume	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
9	Resume	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
10	Resume	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
11	Resume	Yes	RW – D=0	Always 1	Always 0	Always 0	RW – D=0
12	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
13	Core	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
14	Resume	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
15	Resume	No	Always 1	Always 1	Always 0	Always 0	RW – D=0
16	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	Always 0	Always 0
17	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	Always 0	Always 0
18	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	RW – D=1	Always 0
19	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	RW – D=0	Always 0
20	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	Always 0	Always 0
21	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=1	Always 0	Always 0
22	Unimplemented		Always 0	Always 0	Reserved 0	Always 0	Reserved 0
23	Core	Yes	1 when GPIO, 0 when IRQ	0 when GPIO, 1 when IRQ	RW – D=0	Always 0	Always 0
24	Resume	Yes	1 when GPIO, 0 when IRQ	1 when IRQ, 0/1 when GPIO	RW – D=1	Always 0	Always 0
25	Resume	Yes	1 when GPIO, 0 when IRQ	1 when IRQ, 0/1 when GPIO	RW – D=1	RW – D=0	Always 0
26	Core	Yes	RW – D=0	Always 1	Always 1	Always 0	Always 0

Table 22-2. GPIO Summary Table (Sheet 2 of 2)

GPIO#	PowerWell	Muxed	GPIO_USE_SEL	GP_IO_SEL	GP_LVL	GPO_BLINK	GPI_INV
27:28	Resume	Yes	1 when GPIO, 0 when IRQ	1 when IRQ, 0/1 when GPIO	RW – D=1	RW – D=0	Always 0
29	Core	Yes	RW – D = 0	Always 1	Always 1	Always 0	Always 0
30:31	Core	Yes	1 when GPIO, 0 when IRQ	Always 1	Always 1	Always 0	Always 0
32	Unimplemented		Always 0	Reserved 0	Reserved 0	N/A	N/A
33:34	Core	Yes	1 when GPIO, 0 when IRQ	1 when IRQ, 0/1 when GPIO	RW – D=1	N/A	N/A
35:39	Unimplemented		Always 0	Always 0	Always 0	N/A	N/A
40	Core	Yes	1 when GPIO, 0 when IRQ	Always 1	Always 0	N/A	N/A
41	Core	Yes	RW – D = 0	Always 1	Always 1	N/A	N/A
42:47	Unimplemented		Always 0	Always 0	Always 0	N/A	N/A
48	Core	No	Always 1	Always 0	RW – D=1	N/A	N/A
49	Unimplemented		Always 0	Always 0	Always 0	N/A	N/A
50:63	Unimplemented		Always 0	Always 0	Always 0	N/A	N/A

Notes:
1. N/A - Not Applicable, since the relevant registers bit do not exist for these GPIO.
2. Reserved - The bit access type may be RW but it is a reserved bit.
3. RW – D=1: The bit access type is RW; default value is 1.

22.2 General Purpose I/O-Mapped Configuration Register Details

The control for the general purpose I/O signals is handled through separate 64-byte I/O space. The General Purpose I/O configuration registers are mapped into I/O space using register GBA in the PCI configuration space for device 31, function 0 (see Section 19.2.2.3, “Offset 48h: GBA: GPIO Base Address Register” on page 740).

Note: For the following registers, if a bit is allocated for a GPIO that doesn’t exist, unless otherwise indicated, the bit always reads 0 and values written to that bit have no effect.

Table 22-3. Bus 0, Device 31, Function 0: Summary of General Purpose I/O Configuration Registers Mapped Through GBA BAR IO BAR (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	“Offset 00h: GPIO_USE_SEL1 - GPIO Use Select 1 {31:0} Register” on page 807	Variable
04h	07h	“Offset 04h: GP_IO_SEL1 - GPIO Input/Output Select 1 {31:0} Register” on page 808	E400FFFFh
0Ch	0Fh	“Offset 0Ch: GP_LVL1 - GPIO Level 1 for Input or Output {31:0} Register” on page 809	FF3F0000h
18h	1Bh	“Offset 18h: GPO_BLINK - GPIO Blink Enable Register” on page 810	00040000h
2Ch	2Fh	“Offset 2Ch: GPI_INV - GPIO Signal Invert Register” on page 812	00000000h

Table 22-3. Bus 0, Device 31, Function 0: Summary of General Purpose I/O Configuration Registers Mapped Through GBA BAR IO BAR (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
30h	33h	"Offset 30h: GPIO_USE_SEL2 - GPIO Use Select 2 {63:32} Register" on page 813	Variable
34h	37h	"Offset 34h: GP_IO_SEL2 - GPIO Input/Output Select 2 {63:32} Register" on page 813	00000300h
38h	3Bh	"Offset 38h: GP_LVL2 - GPIO Level for Input or Output 2 {63:32} Register" on page 814	00030207h

22.2.1 Register Descriptions

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables" on page 183).

22.2.1.1 Offset 00h: GPIO_USE_SEL1 -GPIO Use Select 1 {31:0} Register

Table 22-4. Offset 00h: GPIO_USE_SEL1 - GPIO Use Select 1 {31:0} Register

Description: This register is used to select between GPIO and alternative functions on GPIO[31:0]					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GPIO_USE_SEL	Enables GPIO[n] (where n is the bit number) to be used as a GPIO, rather than for the alternative function. 1 = Signal used as GPIO (or unmuxed). 0 = Signal used as alternative function. • Bit Access is always RO and returns 0 for bit[22] since there is no GPIO pin[22]. • Bit Access is always RO and returns 1 for bit[0:1, 6:10, 12:15] since there is no alternative function for GPIO pins[0:1, 6:10, 12:15]. • Bit Reset Value is 0000_F7C3h when SIU2_TXD is strapped low on the rising edge of PWROK, else DBBFF7C3h. • See Section 22.1, "Overview" on page 803 for more details. • See Table 22-1 for list of alternate functions.		Variable	RW

a. Core for 0:7, 16:21, 23; Resume for 8:15, 24:31.

22.2.1.2 Offset 04h: GP_IO_SEL1 - GPIO Input/Output Select 1 {31:0} Register

Table 22-5. Offset 04h: GP_IO_SEL1 - GPIO Input/Output Select 1 {31:0} Register

Description: This register allows setting of input/output direction of the GPIO pins 31-0					
View: PCI	**BAR:** GBA(IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** E400FFFFh			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	GPI_31_29	Always 1. The GPI pins are always inputs.		111b	RO
28 : 27	GP_IO_SEL_28_27	• Input when in IRQ mode. I/O when in GPIO mode. 0 = The GPIO signal is programmed as an output. 1 = The corresponding GPIO signal (if enabled in Table 22-4, "Offset 00h: GPIO_USE_SEL1 - GPIO Use Select 1 {31:0} Register" on page 807) is programmed as an input.		00b	RW
26	GPI_26	Always 1. The GPI pins are always inputs.		1b	RO
25 : 24	GP_IO_SEL_25_24	• Input when in IRQ mode. I/O when in GPIO mode. 0 = The GPIO mode signal is programmed as an output. 1 = The corresponding GPIO mode signal (if enabled in Table 22-4, "Offset 00h: GPIO_USE_SEL1 - GPIO Use Select 1 {31:0} Register" on page 807) is programmed as an input.		00b	RW
23	GP_IO_23	This GPIO pin is output, when in GPIO mode (reports 0h). Input when used as IRQ (reports 1h)		0h	RO
22	Reserved	Reserved. No corresponding GPIO.		0h	RO
21 : 16	GP_IO_21_16	These GPIO pins are outputs, when in GPIO mode (reports 00h). Input when used as IRQ (reports 3Fh)		00h	RO
15 : 00	GPI_15_0	Always 1. The GPI pins are always inputs.		FFFFh	RO

a. Core for 0:7, 16:21, 23; Resume for 8:15, 24:31.

22.2.1.3 Offset 0Ch: GP_LVL1 - GPIO Level 1 for Input or Output {31:0} Register

Table 22-6. Offset 0Ch: GP_LVL1 - GPIO Level 1 for Input or Output {31:0} Register (Sheet 1 of 2)

Description: This register allows reading of the current GPIO bit values for GPIO pins 31-0 when input, and writing the value when output.					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0Ch **Offset End:** 0Fh	
Size: 32 bit	**Default:** FF3F0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	GP_LVL_31_29	These bits correspond to input-only GPIO in the core well. The corresponding GP_LVL bit reflects the state of the input signal. Writes to these bits have no effect. 0 = Low 1 = High These bits correspond to GPIO that are in the core well and are reset to their native function by RSMRST#.		111b	RO
28 : 27	GP_LVL_27_28	If GPIO[n] is programmed to be an output (via the corresponding bit in the GP_IO_SEL register), then the corresponding GP_LVL[n] bit can be updated by software to drive a high or low value on the output pin. If GPIO[n] is programmed as an input, then the corresponding GP_LVL bit reflects the state of the input signal. Writes have no effect. 0 = Low 1 = High These bits correspond to GPIO that are in the Resume well and are reset to their native function by RSMRST# and by a writing to the CF9h register.		11b	RW
26	GP_LVL_26	This bit corresponds to input-only GPI in the core well. The corresponding GP_LVL bit reflects the state of the input signal. Writes to this bit have no effect. 0 = Low 1 = High This bit corresponds to a GPI that is in the core well and is reset to its native function by RSMRST#.		1b	RO
25 : 24	GP_LVL_25_24	If GPIO[n] is programmed to be an output (via the corresponding bit in the GP_IO_SEL register), then the corresponding GP_LVL[n] bit can be updated by software to drive a high or low value on the output pin. If GPIO[n] is programmed as an input, then the corresponding GP_LVL bit reflects the state of the input signal. Writes have no effect. 0 = Low 1 = High These bits correspond to GPIO that are in the Resume well and are reset to their native function by RSMRST# and by a writing to the CF9h register.		11b	RW
23	GP_LVL_23	The bit can be updated by software to drive a high or low value on the output pin when used as GPIO function. 0 = Low 1 = High The corresponding GPIO pin is an input when used as IRQ. This bit correspond to GPIO that is in the core well and is reset to its native function by PLTRST#.		0h	RW

Table 22-6. Offset 0Ch: GP_LVL1 - GPIO Level 1 for Input or Output {31:0} Register (Sheet 2 of 2)

Description: This register allows reading of the current GPIO bit values for GPIO pins 31-0 when input, and writing the value when output.					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 0Ch **Offset End:** 0Fh	
Size: 32 bit	**Default:** FF3F0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
22	Reserved	Reserved. No corresponding GPIO.		0h	RW
21 : 16	GP_LVL_21_16	These bits can be updated by software to drive a high or low value on the output pin when used as GPIO functions. 0 = Low 1 = High The corresponding GPIO pins are input when used as IRQ. These bits correspond to GPIOs that are in the core well and are reset to their native function by PLTRST#.		3Fh	RW
15 : 00	Reserved	Reserved.		0000h	RO

a. Core for 0:7, 16:21, 23; Resume for 8:15, 24:31.

22.2.1.4 Offset 18h: GPO_BLINK - GPIO Blink Enable Register

Table 22-7. Offset 18h: GPO_BLINK - GPIO Blink Enable Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 18h **Offset End:** 1Bh	
Size: 32 bit	**Default:** 00040000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :29	Reserved	Reserved.		0h	RO
28 : 27	GPO_BLINK_28_27	The setting of this bit has no effect if the corresponding GPIO signal is programmed as an input. 0 = The corresponding GPIO functions normally. 1 = If the corresponding GPIO is programmed as an output, the output signal blinks at a rate of approximately once per second. The high and low times have approximately 0.5 seconds each. The GP_LVL bit is not altered when this bit is set. The usage model for a blinking output is to control an LED. This value does not need to have exactly one second granularity, but must be close. The value of the corresponding GP_LVL bit remains unchanged during the blink process, and does not effect the blink in any way. The GP_LVL bit is not altered when programmed to blink. It remains at its previous value. These bits correspond to GPIO in the Resume well and are reset to their native function by RSMRST# or a write to the CF9h register or any other PLTRST#		00b	RW
26	Reserved	Reserved.		0h	RO

Table 22-7. Offset 18h: GPO_BLINK - GPIO Blink Enable Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** GBA(IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 18h **Offset End:** 1Bh	
Size: 32 bit	**Default:** 00040000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
25	GPO_BLINK_25	The setting of this bit has no effect if the corresponding GPIO signal is programmed as an input. 0 = The corresponding GPIO functions normally. 1 = If the corresponding GPIO is programmed as an output, the output signal blinks at a rate of approximately once per second. The high and low times have approximately 0.5 seconds each. The GP_LVL bit is not altered when this bit is set. The usage model for a blinking output is to control an LED. This value does not need to have exactly one second granularity, but must be close. The value of the corresponding GP_LVL bit remains unchanged during the blink process, and does not effect the blink in any way. The GP_LVL bit is not altered when programmed to blink. It remains at its previous value. These bits correspond to GPIO in the Resume well and are reset to their native function by RSMRST# or a write to the CF9h register or any other PLTRST#.		0b	RW
24 :20	Reserved	Reserved		0h	RO
19 : 18	GPO_BLINK_19_18	The setting of this bit has no effect if the corresponding GPIO signal is programmed as an input. 0 = The corresponding GPIO functions normally. 1 = If the corresponding GPIO is programmed as an output, the output signal blinks at a rate of approximately once per second. The high and low times are approximately 0.5 seconds each. The GP_LVL bit is not altered when this bit is set. The usage model for a blinking output is to control an LED. This value does not need to have exactly one second granularity, but must be close. The value of the corresponding GP_LVL bit remains unchanged during the blink process, and does not effect the blink in any way. The GP_LVL bit is not altered when programmed to blink. It remains at its previous value. These bits correspond to GPIO in the core well and are reset to their native function by PLTRST#.		01b	RW
17 :00	Reserved	Reserved.		0h	RO

a. Core for 0:7, 16:21, 23; Resume for 8:15, 24:31.

22.2.1.5 Offset 2Ch: GPI_INV - GPIO Signal Invert Register

Table 22-8. Offset 2Ch: GPI_INV - GPIO Signal Invert Register

Description: This register allows GPIO inputs for GPIO pins 31-0 to be inverted.					
View: PCI	**BAR:** GBA(IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		00h	RO
15 : 14	GPI_INV_15_14	**Input Inversion:** These bits only have effect if the corresponding GPIO is used as an input. This is used to allow active-low and active-high inputs to cause SMI# or SCI. 0 = No bit is inverted. 1 = The corresponding data value in the GP_LVL bit is inverted. These bits correspond to GPI in the resume well and are reset to their native function by RSMRST# or a write to the CF9h register or any other PLTRST#. For triggering requirements, see Section 22.3.2, "Triggering" on page 815. The setting of these bits have no effect if the corresponding GPIO is programmed as an output.		00b	RW
13 : 12	GPI_INV_13_12	**Input Inversion**: These bits only has effect if the corresponding GPIO is used as an input. This is used to allow active-low and active-high inputs to cause SMI# or SCI. 0 = No bit is inverted. 1 = The corresponding data value in the GP_LVL bit is inverted. These bits correspond to GPI in the core well and are reset to their native function by RSMRST#.		00b	RW
11 : 08	GPI_INV_11_8	**Input Inversion:** These bits only have effect if the corresponding GPIO is used as an input. This is used to allow active-low and active-high inputs to cause SMI# or SCI. 0 = No bit is inverted. 1 = The corresponding data value in the GP_LVL bit is inverted. These bits correspond to GPI in the resume well and are reset to their native function by RSMRST# or a write to the CF9h register or any other PLTRST#. For triggering requirements, see Section 22.3.2, "Triggering" on page 815. The setting of these bits have no effect if the corresponding GPIO is programmed as an output.		0h	RW
07 : 00	GPI_INV_7_0	**Input Inversion**: This bit only has effect if the corresponding GPIO is used as an input. This is used to allow active-low and active-high inputs to cause SMI# or SCI. 0 = No bit is inverted. 1 = The corresponding data value in the GP_LVL bit is inverted. These bits correspond to GPI in the core well and are reset to their native function by PLTRST#.		00h	RW

a. Core for 0:7, 16:21, 23; Resume for 8:15, 24:31.

22.2.1.6 Offset 30h: GPIO_USE_SEL2 - GPIO Use Select 2 {63:32} Register

Table 22-9. Offset 30h: GPIO_USE_SEL2 - GPIO Use Select 2 {63:32} Register

Description: This register is used to select between GPIO and alternative functions on GPIO[63:32]					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 30h **Offset End:** 33h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GPIO_USE_SEL	Enables GPIO[n] (where n is the bit number) to be used as a GPIO, rather than for the alternative function. 1 = Signal used as GPIO (or unmuxed). 0 = Signal used as alternative function. • Bit Access is always RO for bits[03:07, 10:15, 18:31] and returns 0 since GPIO pins[35:39, 42:47, 50:63] do not exist. • Bit Access is always RO for bit[16] and returns 1 since there is no alternative function for GPIO[48]. • Bit Reset Value is 0001_0000h when SIU2_TXD is strapped LOW on the rising edge of PWROK, else 0001_0106h. • See Section 22.1, “Overview” on page 803 for more details. • See Table , “” on page 804 for list of alternate functions.		Variable	RW

22.2.1.7 Offset 34h: GP_IO_SEL2 - GPIO Input/Output Select 2 {63:32} Register

Table 22-10. Offset 34h: GP_IO_SEL2 - GPIO Input/Output Select 2 {63:32} Register (Sheet 1 of 2)

Description: This register allows setting of input/output direction of the GPIO pins 49-32					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 34h **Offset End:** 37h	
Size: 32 bit	**Default:** 00000300h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 17	GP_IO_SEL_63_49	Always 0. These pins are always outputs.		0h	RO
16 : 16	GP_IO_SEL_48	Always 0. These pins are always outputs.		0h	RO
15 : 10	GP_IO_SEL_47_42	Always 0. No corresponding GPIO.		0h	RO
09 : 08	GP_IO_SEL_41_40	Always 1. These pins are always inputs.		11b	RO

Table 22-10. Offset 34h: GP_IO_SEL2 - GPIO Input/Output Select 2 {63:32} Register (Sheet 2 of 2)

Description: This register allows setting of input/output direction of the GPIO pins 49-32					
View: PCI	**BAR:** GBA(IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 34h **Offset End:** 37h	
Size: 32 bit	**Default:** 00000300h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	GP_IO_SEL_39_35	Always 0. No corresponding GPIO.		0h	RO
02 : 01	GP_IO_SEL_34_33	• Input when in IRQ mode. I/O when in GPIO mode. 0 = GPIO signal is programmed as an output. 1 = Corresponding GPIO signal is programmed as an input (if enabled in Table 22-9, "Offset 30h: GPIO_USE_SEL2 - GPIO Use Select 2 {63:32} Register" on page 813).		00b	RW
00	Reserved	Reserved. No corresponding GPIO.		0b	RW

22.2.1.8 Offset 38h: GP_LVL2 - GPIO Level for Input or Output 2 {63:32} Register

Table 22-11. Offset 38h: GP_LVL2 - GPIO Level for Input or Output 2 {63:32} Register (Sheet 1 of 2)

Description: This register allows reading of the current GPIO bit values for GPIO pins 31-0 when input, and writing the value when output.					
View: PCI	**BAR:** GBA(IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 38h **Offset End:** 3Bh	
Size: 32 bit	**Default:** 00030207h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 17	Reserved	Read-Only 0.		0h	
16 : 16	GP_LVL_48	The corresponding GP_LVL[n] bit can be updated by software to drive a high or low value on the output pin. 0 = low 1 = high Since these bits correspond to GPIO that are in the processor I/O and core well, respectively, these bits are reset to their native function by PLTRST#.		1b	RW
15 : 10	Reserved	Read-Only 0.		0h	
09 : 08	GP_LVL_41_40	The corresponding GP_LVL[n] bit reflects the state of the input signal. Writes have no effect. 0 = low 1 = high Since these bits correspond to GPIO that are in the core well they are reset to their native function by PLTRST#.		10b	RO

Table 22-11. Offset 38h: GP_LVL2 - GPIO Level for Input or Output 2 {63:32} Register (Sheet 2 of 2)

Description: This register allows reading of the current GPIO bit values for GPIO pins 31-0 when input, and writing the value when output.					
View: PCI	**BAR:** GBA(IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 38h **Offset End:** 3Bh	
Size: 32 bit	**Default:** 00030207h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Read-Only 0.		0h	
02 : 01	GP_LVL_34_33	If GPIO[n] is programmed to be an output (via the corresponding bit in the GP_IO_SEL register), then the corresponding GP_LVL[n] bit can be updated by software to drive a high or low value on the output pin. If GPIO[n] is programmed as an input, then the corresponding GP_LVL bit reflects the state of the input signal (1 = high, 0 = low). Writes have no effect. 0 = low 1 = high Since these bits correspond to GPIO that are in the core well and are reset to their native function by PLTRST#.		11b	RW
00	Reserved	Reserved. No corresponding GPIO.		1b	RW

22.3 Additional GPIO Theory of Operation

22.3.1 SMI# and SCI Routing

The routing bits for GPIO[0:15] allow an input to be routed to SMI# or SCI, or neither. See Chapter 40, “Power Management” for the routing register. A bit can be routed to either an SMI# or an SCI, but not both.

22.3.2 Triggering

GPIO[0:15] have “sticky” bits on the input. See Chapter 40, “Power Management” for the GPE0_STS register and the ALT_GPI_SMI_STS register. As long as the signal goes active for at least two clocks (PCI clock while in S0-S1 state, RTC clock while in S3-S5 state),CMI keeps the sticky status bit active. The active level (high or low) can be selected via the GP_INV register.

If the system is in an S0 or S1-D state, the GPI are sampled at 33 MHz, so the signal only needs to be active for about 60 ns to be latched. In S3,-S5 states, the GPI are sampled at 32.768 kHz, and thus must be active for at least 61 µs to be latched.

Note: GPIs that are in the core well are not capable of waking the system from sleep states where the core well is not powered.

If the input signal is still active when the latch is cleared, it is again set (another edge is not required). This makes these signals “level” triggered inputs.

§ §

23.0 SATA: Bus 0, Device 31, Function 2

23.1 SATA PCI Configuration Registers

All of the SATA configuration registers are in the core well. All registers not mentioned are reserved.

Start	End	Functionality Described
00	3F	PCI Header
40	5F	SFF-8038i Configuration (legacy bus master IDE)
70	7F	PCI Power Management Capability Pointer
80	8F	Message Signaled Interrupt Capability Pointer
90	A7	Additional Configuration (test modes, etc.)
A8	AF	Serial ATA Capability Pointer
B0	FF	Additional Configuration (test modes, etc.)

Table 23-1. Bus 0, Device 31, Function 2: Summary of SATA Controller PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: ID – Identifiers Register" on page 819	Variable
04h	05h	"Offset 04h: CMD - Command Register" on page 819	0000h
06h	07h	"Offset 06h: STS - Device Status Register" on page 820	02B0h
08h	08h	"Offset 08h: RID - Revision ID Register" on page 821	Variable
0Ah	0Bh	"Offset 0Ah: CC - Class Code Register" on page 823	Variable
0Dh	0Dh	"Offset 0Dh: MLT – Master Latency Timer Register" on page 823	00h
10h	13h	"Offset 10h: PCMDBA – Primary Command Block Base Address Register" on page 824	00000001h
14h	17h	"Offset 14h: PCTLBA – Primary Control Block Base Address Register" on page 824	00000001h
18h	1Bh	"Offset 18h: SCMDBA – Secondary Command Block Base Address Register" on page 825	00000001h
1Ch	1Fh	"Offset 1Ch: SCTLBA – Secondary Control Block Base Address Register" on page 825	00000001h
20h	23h	"Offset 20h: LBAR – Legacy Bus Master Base Address Register when SCC is SATA with AHCI PI" on page 826	00000001h
24h	27h	"Offset 24h: ABAR – AHCI Base Address Register" on page 826	00000000h
2Ch	2Fh	"Offset 2Ch: SS - Sub System Identifiers Register" on page 827	00000000h
34h	34h	"Offset 34h: CAP – Capabilities Pointer Register" on page 827	80h
3Ch	3Dh	"Offset 3Ch: INTR - Interrupt Information Register" on page 828	Variable

Table 23-1. Bus 0, Device 31, Function 2: Summary of SATA Controller PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
40h	41h	"Offset 40h: PTIM – Primary Timing Register" on page 829	0000h
44h	44h	"Offset 44h: D1TIM – Device 1 IDE Timing Register" on page 830	00h
48h	48h	"Offset 48h: SYNCC – Synchronous DMA Control Register" on page 831	00h
4Ah	4Bh	"Offset 4Ah: SYNCTIM – Synchronous DMA Timing Register" on page 832	0000h
54h	57h	"Offset 54h: IIOC – IDE I/O Configuration Register" on page 833	00000000h
70h	71h	"Offset 70h: PID – PCI Power Management Capability ID Register" on page 834	Variable
72h	73h	"Offset 72h: PC – PCI Power Management Capabilities Register" on page 834	4002h
74h	77h	"Offset 74h: PMCS – PCI Power Management Control And Status Register" on page 835	0000h
80h	81h	"Offset 80h: MID – Message Signaled Interrupt Identifiers Register" on page 836	7005h
82h	83h	"Offset 82h: MC – Message Signaled Interrupt Message Control Register" on page 837	0000h
84h	87h	"Offset 84h: MA – Message Signaled Interrupt Message Address Register" on page 838	00000000h
88h	89h	"Offset 88h: MD – Message Signaled Interrupt Message Data Register" on page 838	0000h
90h	90h	"Offset 90h: MAP – Port Mapping Register" on page 839	00h
92h	92h	"Offset 92h: PCS – Port Control and Status Register" on page 840	00h
A8h	ABh	"Offset A8h: SATACR0 – Serial ATA Capability Register 0" on page 841	00100012h
ACh	AFh	"Offset ACh: SATACR1 – Serial ATA Capability Register 1" on page 841	00000048h
C0h	C0h	"Offset C0h: ATC – APM Trapping Control Register" on page 842	00h
C4h	C4h	"Offset C4h: ATS – ATM Trapping Status Register" on page 843	00h
D0h	D3h	"Offset D0h: SP – Scratch Pad Register" on page 844	00000000h
E0h	E3h	"Offset E0h: BFCS – BIST FIS Control/Status Register" on page 844	00000000h
E4h	E7h	"Offset E4h: BFTD1 – BIST FIS Transmit Data 1 Register" on page 846	00000000h
E8h	EBh	"Offset E8h: BFTD2 – BIST FIS Transmit Data 2 Register" on page 846	0h
F8h	FBh	"Offset F8h: MANID – Manufacturing ID Register" on page 847	Variable

23.1.1 PCI Header

The default values are defined with an h for hex, a b for binary, or 00 for zero. If there is not a a letter showing the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values and are read only. Writes to reserved locations may cause system failure and unpredictable results.

Note: Reserved bits are read only.

23.1.1.1 Offset 00h: ID - Identifiers Register

Table 23-2. Offset 00h: ID – Identifiers Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	DID	**Device ID (DID):** The value reported in this field is in the range between 5028-502Bh. The specific value is dependent on MAP.SMS, MAP.MV.		Variable	RO
15 : 00	VID	**Vendor ID (VID):** 16-bit field which indicates the company vendor as Intel.		8086h	RO

23.1.1.2 Offset 04h: CMD - Command Register

Table 23-3. Offset 04h: CMD - Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RO
10	ID	**Interrupt Disable (ID):** This disables pin-based INTx# interrupts. This bit has no effect on MSI operation. When set, internal INTx# messages will not be generated. When cleared, internal INTx# messages are generated if there is an interrupt and MSI is not enabled.		0h	RW
09	FBE	Fast Back-to-Back Enable (FBE): Reserved.		0h	RO
08	SEE	SERR# Enable (SEE): Reserved. The SATA Controller never generates an SERR#.		0h	RO
07	WCC	Wait Cycle Enable (WCC): Reserved.		0h	RO
06	PEE	**Parity Error Response Enable (PEE):** When set, the SATA Controller will corrupt the outbound DATA FIS CRC if a forwarded data parity error is indicated.		0h	RW
05	VGA	VGA Palette Snooping Enable (VGA): Reserved		0h	RO

Table 23-3. Offset 04h: CMD - Command Register (Sheet 2 of 2)

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 04h Offset End: 05h	
Size: 16 bit	Default: 0000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	MWIE	Memory Write and Invalidate Enable (MWIE): Reserved		0h	RO
03	SCE	Special Cycle Enable (SCE): Reserved		0h	RO
02	BME	**Bus Master Enable (BME):** Controls the SATA Controller's ability to act as a master for data transfers. This bit does not impact the generation of completions for split transaction commands.		0h	RW
01	MSE	**Memory Space Enable (MSE):** Controls access to the SATA Controller's target memory space (for AHCI).'		0h	RW
00	IOSE	**I/O Space Enable (IOSE):** Controls access to the SATA Controller's target I/O space.		0h	RW

23.1.1.3 Offset 06h: STS - Device Status Register

Table 23-4. Offset 06h: STS - Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 06h Offset End: 07h	
Size: 16 bit	Default: 02B0h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error (DPE):** Set when the SATA Controller detects a parity error on its interface.		0h	RWC
14	SSE	**Signaled System Error (SSE):** The SATA Controller will never generate an SERR#.		0h	RO
13	RMA	**Received Master-Abort Status (RMA):** Set when the SATA Controller receives a master abort to a cycle it generated.		0h	RWC
12	RTA	**Received Target-Abort Status (RTA):** Set when the SATA Controller receives a target abort to a cycle it generated.		0h	RWC
11	STA	**Signaled Target-Abort Status (STA):** Reserved. The SATA Controller will never generate a target abort.		0h	RO
10 : 09	DEVT	**DEVSEL# Timing Status (DEVT):** Controls the device select time for the SATA Controller's PCI interface.		01h	RO
08	DPD	**Master Data Parity Error Detected (DPD):** Set when the SATA Controller, as a master, either detects a parity error or sees the parity error line asserted, and the parity error response bit (bit 6 of the command register) is set. This bit can only be set on read completions received from the backbone where there is a parity error.		0h	RWC
07	Reserved	Fast Back-to-Back Capable: Reserved		1h	RO
06	Reserved	Reserved		0h	RO
05	C66	66 MHz Capable		1h	RO

Table 23-4. Offset 06h: STS - Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 02B0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	CL	**Capabilities List (CL):** Indicates the presence of a capabilities list. The minimum requirement for the capabilities list must be PCI power management for the SATA Controller.		1h	RO
03	IS	**Interrupt Status (IS):** Reflects the state of INTx# messages. This bit is set when the interrupt is to be asserted. This bit is a 0 after the interrupt is cleared (independent of the state of CMD.ID).		0h	RO
02 : 00	Reserved	Reserved		0h	RO

23.1.1.4 Offset 08h: RID - Revision ID Register

Table 23-5. Offset 08h: RID - Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision ID (RID):** Indicates stepping of the host controller hardware. This register follows the ICH revision ID scheme as defined in Section 19.2.1.4, “Offset 08h: RID - Revision ID Register” on page 736.		Variable	RO

23.1.1.5 PI - Programming Interface Register

Table 23-6. Programming Interface, DID and CC.SCC Register Value Definitions

Input Parameters		Resulting Values		
MAP.SMS: 00 - Select IDE 01 - Select AHCI 10 - Reserved 11 - Reserved	MAP.MV	CC.SCC	PI	DID
00b	00b	01h (IDE)	8Ah[a] (RW)	5028h
01b	00b	06h (SATA)	01h (RO)	5029h
10b	00b	Illegal[b]	Illegal	Illegal
11b	00b	Illegal	Illegal	Illegal

a. PI register bits 2 and 0 are write-able at all times when MAP.MV=00h (so the value of this field may be 8Ah, 8Bh, 8Dh, or 8Fh), although the write only affects the SATA controller's operation when CC.SCC is 01h. If PI is written while CC.SCC is 04h, the value shown will not be 8Ah when CC.SCC is changed from 04h to 01h, but rather the value of the last write to PI. BIOS must always check and set PI to the desired mode of operation (legacy or native) when CC.SCC is 01h.

b. "Illegal" means that SW must not program this combination otherwise the results are undefined.

23.1.1.5.1 Programming Interface when CC.SCC = “01h”

Table 23-7. Programming Interface when CC.SCC = “01h”

Bit	Type	Reset	Description
7	RO	1	Indicates the SATA Controller supports bus master operation.
6:4	RO	0	Reserved
3	RO	1	**Secondary Mode Native Capable (SNC):** Indicates that the secondary controller supports both legacy and native modes.
2	RW	0	**Secondary Mode Native Enable (SNE):** Determines the mode that the secondary channel is operating in. ‘0’ corresponds to 'compatibility', ‘1’ means PCI native. If this bit is set by SW, then the PNE bit must also be set by SW. While in theory these bits can be programmed separately, such a configuration is not supported by today’s software and is not supported by this hardware.
1	RO	1	**Primary Mode Native Capable (PNC):** Indicates that the primary controller supports both legacy and native modes.
0	RW	0	**Primary Mode Native Enable (PNE):** Determines the mode that the primary channel is operating in. ‘0’ corresponds to 'compatibility', ‘1’ means PCI native. If this bit is set by SW, then the SNE bit must also be set by SW. While in theory these bits can be programmed separately, such a configuration is not supported by today’s software and is not supported by this hardware.

23.1.1.5.2 Programming Interface when CC.SCC = "06h"

Table 23-8. Programming Interface when CC.SCC = "06h"

Bit	Type	Reset	Description
07:00	RO	01h	**Interface (IF):** Indicates the SATA Controller is AHCI 1.0 compliant. Internally, under this condition, the SATA controller is in native mode and its I/O spaces are only accessible through the I/O BARs.

23.1.1.6 Offset 0Ah: CC - Class Code Register

Table 23-9. Offset 0Ah: CC - Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 0Ah **Offset End:** 0Bh	
Size: 16 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	BCC	**Base Class Code (BCC):** Indicates that this is a mass storage device.		01h	RO
07 : 00	SCC	**Sub Class Code (SCC):** The value reported in this field is dependent on MAP.SMS, MAP.MV, . See the table in section 48.1.1.5.		Variable	RO

23.1.1.7 Offset 0Dh: MLT – Master Latency Timer Register

Table 23-10. Offset 0Dh: MLT – Master Latency Timer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MLT	**Master Latency Timer (MLT):** This register has no meaning as the controller lives on NSI.		00h	RO

23.1.1.8 Offset 10h: PCMDBA – Primary Command Block Base Address Register

This 8-byte I/O space is used in Native Mode for the Primary Controller's Command Block.

Table 23-11. Offset 10h: PCMDBA – Primary Command Block Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 03	BAR	**Base Address (BAR):** Base address of the I/O space (8 consecutive I/O locations).		0h	RW
02 : 01	Reserved	Reserved		0h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for IO space.		1h	RO

23.1.1.9 Offset 14h: PCTLBA – Primary Control Block Base Address Register

This 4-byte I/O space is used in Native Mode for the Primary Controller's Control Block.

Table 23-12. Offset 14h: PCTLBA – Primary Control Block Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 02	BAR	**Base Address (BAR):** Base address of the I/O space (4 consecutive I/O locations).		0h	RW
01	Reserved	Reserved		0h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for IO space.		1h	RO

23.1.1.10 Offset 18h: SCMDBA – Secondary Command Block Base Address Register

This 8-byte I/O space is used in Native Mode for the Secondary Controller's Command Block.

Table 23-13. Offset 18h: SCMDBA – Secondary Command Block Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 18h **Offset End:** 1Bh	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 03	BAR	**Base Address (BAR):** Base address of the I/O space (8 consecutive I/O locations).		0h	RW
02 : 01	Reserved	Reserved		0h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for IO space		1h	RO

23.1.1.11 Offset 1Ch: SCTLBA – Secondary Control Block Base Address Register

This 4-byte I/O space is used in Native Mode for the Secondary Controller's Control Block.

Table 23-14. Offset 1Ch: SCTLBA – Secondary Control Block Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 1Ch **Offset End:** 1Fh	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 02	BAR	**Base Address (BAR):** Base address of the I/O space (4 consecutive I/O locations).		0h	RW
01	Reserved	Reserved		0h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for IO space.		1h	RO

23.1.1.12 Offset 20h: LBAR – Legacy Bus Master Base Address Register

This BAR is used to allocate I/O space for the SFF-8038i mode of operation (aka Bus Master IDE) and AHCI index/data pair.

Table 23-15. Offset 20h: LBAR – Legacy Bus Master Base Address Register when SCC is SATA with AHCI PI

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 20h Offset End: 23h	
Size: 32 bit	Default: 00000001h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 04	BA	**Base Address (BA):** Base address of the I/O space. • Note on bit 4: When CC.SCC is 01h, bit 4 is RW, resulting in requesting 16B of I/O space. When CC.SCC is not 01h, this bit is RO and reports 0, resulting in requesting 32B of I/O space. • Bits 15:5 are always RW with default value of 0. •		0h	RW
03 : 01	Reserved	Reserved		0h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for I/O space.		1h	RO

23.1.1.13 Offset 24h: ABAR – AHCI Base Address Register

This register allocates space for the memory registers defined in section 1.3.

Note that hardware only supports non-combined mode.

Note that this register must be programmed to value of 0001_0000h or greater, otherwise memory cycles targeting the ABAR range may not accepted.

Table 23-16. Offset 24h: ABAR – AHCI Base Address Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 24h Offset End: 27h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 10	BA	**Base Address (BA):** Base address of register memory space (aligned to 1 KB)		0h	RW
09 : 04	Reserved	Reserved		0h	RO
03	PF	**Prefetchable (PF):** Indicates that this range is not pre-fetchable		0h	RO
02 : 01	TP	**Type (TP):** Indicates that this range can be mapped anywhere in 32-bit address space		00h	RO
00	RTE	**Resource Type Indicator (RTE):** Indicates a request for register memory space.		0h	RO

23.1.1.14 Offset 2Ch: SS - Sub System Identifiers Register

This register is initialized to logic 0 by the assertion of PLTRST#. This register can be written only once after PLTRST# de-assertion.

Table 23-17. Offset 2Ch: SS - Sub System Identifiers Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	SSID	**Subsystem ID (SSID):** This is written by BIOS. No hardware action taken on this value.		0000h	RWO
15 : 00	SSVID	**Subsystem Vendor ID (SSVID):** This is written by BIOS. No hardware action taken on this value.		0000h	RWO

23.1.1.15 Offset 34h: CAP – Capabilities Pointer Register

Table 23-18. Offset 34h: CAP – Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Capability Pointer (CP):** If CC.SCC=01h, then CAP is set to offset 70h to indicate that the first capability pointer is the PCI Power Management capability. If CC.SCC is not set to 01h, then CAP is set to offset 80h to indicate that the first capability pointer is the Message Signaled Interrupt capability.		80h	RO

23.1.1.16 Offset 3Ch: INTR - Interrupt Information Register

Table 23-19. Offset 3Ch: INTR - Interrupt Information Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 3Ch **Offset End:** 3Dh	
Size: 16 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	IPIN	**Interrupt Pin (IPIN):** This reflects the value of D31IP.SIP in chipset configuration space.		Variable	RO
07 : 00	ILINE	**Interrupt Line (ILINE):** Software written value to indicate which interrupt line (vector) the interrupt is connected to. No hardware action is taken on this register.		00h	RW

23.1.2 Additional SFF-8038i Configuration Registers

The following registers are necessary to implement as read/write bits in order to maintain software functionality for the SFF-8038i mode of operation (a.k.a. Bus Master IDE). They have no bearing on Serial ATA operation unless otherwise indicated, but are necessary to be read/write for legacy software capability.

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Start	End	Symbol	Name
40	41	PTIM	Primary IDE Timing
42	43	STIM	Secondary IDE Timing
44	44	D1TIM	Slave IDE Timing
48	48	SYNCC	Synchronous DMA Control Register
4A	4B	SYNCTIM	Synchronous DMA Timing Register
54	57	IIOC	IDE I/O Configuration

23.1.2.1 Offset 40h: PTIM – Primary Timing Register

This controls the timings driven on the parallel cable.

Table 23-20. Offset 40h: PTIM – Primary Timing Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:2		**Offset Start:** 40h **Offset End:** 41h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DE	**Decode Enable (DE):** Enables the SATA Controller to decode the Command Blocks (1F0-1F7h for primary, 170-177h for secondary or their native BAR equivalents) and Control Block (3F6h for primary and 376h for secondary or their native BAR equivalents). This bit still has functionality in SATA – if this bit is not set, the port that is mapped to this range will not be decoded.		0h	RW
14	D1STE	**Device 1 Separate Timing Enable (D1STE):** When cleared, both device 0 and device 1 use the same timings, as defined by bits 13:12 and bits 9:8 of this register. When set, device 0 uses these timings, but device 1 uses the timings from the “Slave Timing” register at offset 44h.		0h	RW
13 : 12	ISP	**IORDY Sample Point (ISP):** Determines the number of 33 MHz clocks between IDE IOR#/IOW# assertion and the first IORDY sample point. 00 = 5 clocks 01 = 4 clocks 10 = 3 clocks 11 = Reserved		00h	RW
11 : 10	Reserved	Reserved		00h	RO
09 : 08	RCT	**Recovery Time (RCT):** The setting of these bits determines the minimum number of 33 MHz clocks between the last IORDY sample point and the IOR#/IOW# strobe of the next cycle. 00 = 4 clocks 01 = 3 clocks 10 = 2 clocks 11 = 1 clock		00h	RW
07	DTE1	**Device 1 DMA Timing Enable (DTE1):** When this bit is set, the fast timing mode is enabled for DMA transfers only for this drive. PIO transfers to the data port will run in compatible timing.		0h	RW
06	PPE1	**Device 1 Prefetch/Posting Enable (PPE1):** When this bit is set, prefetch and posting to the IDE data port is enabled for this drive.		0h	RW
05	E1	**Device 1 IORDY Sample Point Enable (IE1):** When this bit is set, IORDY sampling will be enabled for this drive. When this bit is cleared, IORDY sampling is disabled for this drive.		0h	RW
04	TIM1	**Device 1 Fast Timing Bank (TIM1):** When cleared, accesses to the data port will use compatible timings for this drive. When set and bit 14 cleared, accesses to the data port will use bits 13:12 for the IORDY sample point, and bits 9:8 for the recovery time. When set and bit 14 set, accesses to the data port will use the IORDY sample point and recover time specified in the slave IDE timing register.		0h	RW
03	DTE0	**Device 0 DMA Timing Enable (DTE0):** When this bit is set, the fast timing mode is enabled for DMA transfers only for this drive. PIO transfers to the IDE data port will run in compatible timing.		0h	RW

Table 23-20. Offset 40h: PTIM – Primary Timing Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 40h Offset End: 41h	
Size: 16 bit	Default: 0000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	PPE0	**Device 0 Prefetch/Posting Enable (PPE0):** When this bit is set, prefetch and posting to the IDE data port is enabled for this drive.		0h	RW
01	IE0	**Device 0 IORDY Sample Point Enable (IE0):** When this bit is set, IORDY sampling will be enabled for this drive. When this bit is cleared, IORDY sampling is disabled for this drive.		0h	RW
00	TIMO	**Device 0 Fast Timing Bank (TIM0):** When cleared, accesses to the data port will use compatible timings for this drive. When set, accesses to the data port will use bits 13:12 for the IORDY sample point, and bits 9:8 for the recovery time		0h	RW

23.1.2.2 STIM – Secondary Timing Register

See the above register description for Primary IDE Timing.

23.1.2.3 Offset 44h: D1TIM – Device 1 IDE Timing Register

The values in this register are only valid if the “Separate Timing Enable” bit is set in the timing registers. Bits 7:4 are used by device 1 on secondary if bit 14 of offset 42h is set, and bits 3:0 are used by device 1 on primary if bit 14 of offset 40h is set.

Table 23-21. Offset 44h: D1TIM – Device 1 IDE Timing Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 44h Offset End: 44h	
Size: 8 bit	Default: 00h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	SISP1	**Secondary Device 1 IORDY Sample Point (SISP1):** Determines the number of 33 MHz clocks between IDE IOR#/IOW# assertion and the first IORDY sample point. 00 = 5 clocks 01 = 4 clocks 10 = 3 clocks 11 = Reserved		00h	RW

Table 23-21. Offset 44h: D1TIM – Device 1 IDE Timing Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 44h **Offset End:** 44h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05 : 04	SRCT1	**Secondary Device 1 Recovery Time (SRCT1):** Determines the minimum number of 33 MHz clocks between the last IORDY sample point and the IOR#/IOW# strobe of the next cycle. 00 = 4 clocks 01 = 3 clocks 10 = 2 clocks 11 = 1 clocks		00h	RW
03 : 02	PISP1	**Primary Device 1 IORDY Sample Point (PISP1):** Same as bits 7:6, except for the primary device.		00h	RW
01 : 00	PRCT1	**Primary Device 1 Recovery Time (PRCT1):** Same as bits 5:4, except for the primary device.		00h	RW

23.1.2.4 Offset 48h: SYNCC – Synchronous DMA Control Register

Table 23-22. Offset 48h: SYNCC – Synchronous DMA Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 48h **Offset End:** 48h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	RO
03	SDAE1	**Secondary Device 1 ATAxx Enable (SDAE1):** When set, enables ATA33/66/100/133 timing modes for the secondary slave device.		0h	RW
02	SDAE0	**Secondary Device 0 ATAxx Enable (SDAE0):** When set, enables ATA33/66/100/133 timing modes for the secondary master device.		0h	RW
01	PDAE1	**Primary Device 1 ATAxx Enable (PDAE1):** When set, enables ATA33/66/100/133 timing modes for the primary slave device.		0h	RW
00	PDAE0	**Primary Device 0 ATAxx Enable (PDAE0):** When set, enables ATA33/66/100/133 timing modes for the primary master device.		0h	RW

23.1.2.5 Offset 4Ah: SYNCTIM – Synchronous DMA Timing Register

The CT and RP values for the ATA_FAST (100MB/s and 133 MB/s modes) are based on 133 MHz clock. The CT and RP values for the 66 MHz and 33 MHz modes are based on either a 66 MHz or 33 MHz clock.

Table 23-23. Offset 4Ah: SYNCTIM – Synchronous DMA Timing Register

<table>
<tr><td>Description:</td><td colspan="5"></td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:31:2</td><td>Offset Start: 4Ah
Offset End: 4Bh</td></tr>
<tr><td>Size: 16 bit</td><td colspan="2">Default: 0000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>15 : 14</td><td>Reserved</td><td>Reserved</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>13 : 12</td><td>SCT1</td><td>Secondary Device 1 Cycle Time (SCT1): The setting of these bits determines the minimum write strobe cycle time (CT) and DMARDY#-to-STOP (RP) time.
<table>
<tr><th rowspan="2">Bits</th><th colspan="2">SBC[3] = 0</th><th colspan="2">SBC[3] = 1</th><th colspan="2">FSBCE[3] = 1</th></tr>
<tr><th>CT</th><th>RP</th><th>CT</th><th>RP</th><th>CT</th><th>RP</th></tr>
<tr><td>00</td><td>4</td><td>6</td><td colspan="2">Reserved</td><td colspan="2">Reserved</td></tr>
<tr><td>01</td><td>3</td><td>5</td><td>3</td><td>8</td><td>3</td><td>16</td></tr>
<tr><td>10</td><td>2</td><td>4</td><td>2</td><td>8</td><td colspan="2">Reserved</td></tr>
<tr><td>11</td><td colspan="2">Reserved</td><td colspan="2">Reserved</td><td colspan="2">Reserved</td></tr>
</table></td><td></td><td>00h</td><td>RW</td></tr>
<tr><td>11 : 10</td><td>Reserved</td><td>Reserved</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>09 : 08</td><td>SCT0</td><td>Secondary Device 0 Cycle Time (SCT0): Same definition as bits 13:12, except for device 0.</td><td></td><td>00h</td><td>RW</td></tr>
<tr><td>07 : 06</td><td>Reserved</td><td>Reserved</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>05 : 04</td><td>PCT1</td><td>Primary Device 1 Cycle Time (PCT1): Same definition as bits 13:12, except for primary device 1.</td><td></td><td>00h</td><td>RW</td></tr>
<tr><td>03 : 02</td><td>Reserved</td><td>Reserved</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>01 : 00</td><td>PCT0</td><td>Primary Device 0 Cycle Time (PCT0): Same definition as bits 13:12, except for primary device 0.</td><td></td><td>00h</td><td>RW</td></tr>
</table>

23.1.2.6 Offset 54h: IIOC – IDE I/O Configuration Register

Table 23-24. Offset 54h: IIOC – IDE I/O Configuration Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:31:2		**Offset Start:** 54h **Offset End:** 57h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		0h	RO
23 : 20	SP2	**Scratchpad (SP2):** No hardware action taken on these bits.		0h	RW
19 : 18	SSM	**Secondary Signals Mode (SSM):** Controls the secondary signals for swap bays. 00 = Normal 01 = Tri-state 10 = Drive Low 11 = Reserved If BUC.PRS in chipset configuration space is '1', then the reset states of these bits will be '01' (tri-state) instead of '00' (normal).		00h	RW
17 : 16	PSM	**Primary Signals Mode (PSM):** Controls the primary signals for swap bays. 00 = Normal 01 = Tri-state 10 = Drive Low 11 = Reserved If BUC.PRS in chipset configuration space is '1', then the reset states of these bits will be '01' (tri-state) instead of '00' (normal).		00h	RW
15 : 12	FSBCE	**Fast Synchronous Base Clock Enable (FSBCE):** When set, enables fast ATA modes. This overrides the state of the SCB[3:0] bits in this register.		0h	RW
11	Reserved	Reserved		0h	RO
10	Reserved	Reserved		0h	RO
09 : 08	Reserved	Reserved – these were the secondary and primary command posting enable bits. These do not exist when parallel ATA is mapped as part of serial ATA, because the command posting BAR has been removed.		0h	RO
07 : 04	SP1	**Scratchpad (SP1):** No hardware action taken on these bits.		0h	RW
03 : 00	SBC	**Synchronous Base Clock (SBC):** Clock used to determine CT and RP timings for synchronous DMA timings. '0' = 33 MHz clock used, '1' = 66 MHz clock used. Bit 3 controls the secondary slave device, bit 2 controls the secondary master device, bit 1 controls the primary slave device, and bit 0 controls the primary master device.		0h	RW

23.1.3 PCI Power Management Capabilities

Start	End	Symbol	Name
70	71	PID	PCI Power Management Capability ID
72	73	PC	PCI Power Management Capabilities
74	77	PMCS	PCI Power Management Control and Status

23.1.3.1 Offset 70h: Offset 70h: PID – PCI Power Management Capability ID Register

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 23-25. Offset 70h: PID – PCI Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 70h **Offset End:** 71h	
Size: 16 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	NEXT	**Next Capability (NEXT):** When CC.SCC is 01h, this field will be 00h indicating this is the last item in the list. When CC.SCC is not 01h, this field will be A8h pointing to the Serial ATA Capability structure.		Variable	RO
07 : 00	CID	**Cap ID (CID):** Indicates that this pointer is a PCI power management.		01h	RO

23.1.3.2 Offset 72h: PC – PCI Power Management Capabilities Register

Table 23-26. Offset 72h: PC – PCI Power Management Capabilities Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 72h **Offset End:** 73h	
Size: 16 bit	**Default:** 4002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	**PME_Support**	Indicates PME# can be generated from the $D3_{HOT}$ state in the SATA controller.		01000b	RO
10	**D2_Support**	The D2 state is not supported.		0h	RO
09	**D1_Support**	The D1 state is not supported.		0h	RO

Table 23-26. Offset 72h: PC – PCI Power Management Capabilities Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 72h **Offset End:** 73h	
Size: 16 bit	**Default:** 4002h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08 : 06	**Aux_Current**	PME# from $D3_{COLD}$ state is not supported, therefore this field is 000b.		000h	RO
05	DSI	**Device Specific Initialization (DSI):** Indicates that no device-specific initialization is required.		0h	RO
04	Reserved	Reserved		0h	RO
03	PMEC	**PME Clock (PMEC):** Indicates that PCI clock is not required to generate PME#.		0h	RO
02 : 00	VS	**Version (VS):** Indicates support for Revision 1.1 of the *PCI Power Management Specification.*		010b	RO

23.1.3.3 Offset 74h: PMCS – PCI Power Management Control And Status Register

Table 23-27. Offset 74h: PMCS – PCI Power Management Control And Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 74h **Offset End:** 77h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PMES	**PME Status (PMES):** This bit is set when a PME event is to be requested, and if this bit is set and PMEE is set, a PME# will be generated.		0h	RWC
14 : 09	Reserved	Reserved – SATA Controller does not implement the data register.		0h	RO
08	PMEE	**PME Enable (PMEE):** When set, the SATA controller generates PME# form $D3_{HOT}$ on a wake event.		0h	RW
07 : 02	Reserved	Reserved		0h	RO
01 : 00	PS	**Power State (PS):** This field is used both to determine the current power state of the SATA Controller and to set a new power state. The values are: 00 = D0 state 11 = $D3_{HOT}$ state When in the $D3_{HOT}$ state, the controller's configuration space is available, but the I/O and memory spaces are not. Additionally, interrupts are blocked. If software attempts to write a '10' or '01' to these bits, the write will be ignored.		00h	RW

23.1.4 Message Signaled Interrupt Capability

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Start	End	Symbol	Name
80	81	MID	Message Signaled Interrupt Capability ID
82	83	MC	Message Signaled Interrupt Message Control
84	87	MA	Message Signaled Interrupt Message Address
88	89	MD	Message Signaled Interrupt Message Data

23.1.4.1 Offset 80h: MID – Message Signaled Interrupt Identifiers Register

Table 23-28. Offset 80h: MID – Message Signaled Interrupt Identifiers Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 80h **Offset End:** 81h	
Size: 16 bit	**Default:** 7005h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	NEXT	**Next Pointer (NEXT):** Indicates the next item in the list is the PCI power management pointer.		70h	RO
07 : 00	CID	**Capability ID (CID):** Capabilities ID indicates MSI.		05h	RO

23.1.4.2 Offset 82h: MC – Message Signaled Interrupt Message Control Register

Table 23-29. Offset 82h: MC – Message Signaled Interrupt Message Control Register

<table>
<tr><td>Description:</td><td colspan="5"></td></tr>
<tr><td>View: PCI</td><td colspan="2">BAR: Configuration</td><td colspan="2">Bus:Device:Function: 0:31:2</td><td>Offset Start: 82h
Offset End: 83h</td></tr>
<tr><td>Size: 16 bit</td><td colspan="2">Default: 0000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>15 : 08</td><td>Reserved</td><td>Reserved</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>07</td><td>C64</td><td>64 Bit Address Capable (C64): Capable of generating a 32-bit message only.</td><td></td><td>0h</td><td>RO</td></tr>
<tr><td>06 : 04</td><td>MME</td><td>Multiple Message Enable (MME): When this field is cleared to ‘000’ (and MSIE is set), only a single MSI message will be generated for all SATA ports, and bits [15:0] of the message vector will be driven from MD[15:0].

<table>
<tr><th rowspan="2">MME</th><th colspan="3">Value Driven on MSI Memory Write</th></tr>
<tr><th>Bits[15:2]</th><th>Bit[1]</th><th>Bit[0]</th></tr>
<tr><td>000</td><td>MD[15:2]</td><td>MD[1]</td><td>MD[0]</td></tr>
<tr><td>001</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>010</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>011–111</td><td colspan="3">Reserved</td></tr>
</table>
Values ‘011b’ to ‘111b’ are reserved. If this field is set to one of these reserved values, the results are undefined.</td><td></td><td>000h</td><td>RW</td></tr>
<tr><td>03 : 01</td><td>MMC</td><td>Multiple Message Capable (MMC): Indicates the number of interrupt message supported by SATA controller.
000: 1 MSI Capable
001: Reserved010: Reserved 100: Reserved</td><td></td><td>000h</td><td>RWO</td></tr>
<tr><td>00</td><td>MSIE</td><td>MSI Enable (MSIE): If set, MSI is enabled and traditional interrupt pins are not used to generate interrupts.</td><td></td><td>0h</td><td>RW</td></tr>
</table>

23.1.4.3 Offset 84h: MA – Message Signaled Interrupt Message Address Register

Table 23-30. Offset 84h: MA – Message Signaled Interrupt Message Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 84h **Offset End:** 87h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	ADDR	**Address (ADDR):** Lower 32 bits of the system specified message address, always DWORD aligned.		0h	RW
01 : 00	Reserved	Reserved		00h	RO

23.1.4.4 Offset 88h: MD – Message Signaled Interrupt Message Data Register

Table 23-31. Offset 88h: MD – Message Signaled Interrupt Message Data Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 88h **Offset End:** 89h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Data (DATA):** This 16-bit field is programmed by system software if MSI is enabled. Its content is driven onto the lower word of the data bus of the MSI memory write transaction.		0h	RW

23.1.5 Additional Configuration Registers

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Start	End	Symbol	Name
90	90	MAP	Port Mapping Register
92	92	PCS	Port Control and Status
A8	AB	SATACR0	Serial ATA Capability Register 0
AC	AF	SATACR1	Serial ATA Capability Register 1

C0	C0	ATC	APM Trapping Control
C4	C4	ATS	APM Trapping Status
D0	D3	SP	Scratch Pad
E0	E3	BFCS	BIST FIS Control/Status
E4	E7	BFTD1	BIST FIS Transmit Data, DW1
E8	EB	BFTD2	BIST FIS Transmit Data, DW2
F8	FB	MANID	Manufacturer's ID

23.1.5.1 Offset 90h: MAP – Port Mapping Register

Table 23-32. Offset 90h: MAP – Port Mapping Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** 90h **Offset End:** 90h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	SMS	**SATA Mode Select (SMS):** SW programs these bits to control the mode in which the SATA HBA should operate: 00b = IDE mode 01b = AHCI mode 10b = Reserved 11b = Reserved ***Notes:*** • IDE mode can be selected when AHCI is enabled. • "Programming these bits with values that are illegal will result in indeterministic behavior by the HW. —		00h	RW
05 : 02	Reserved	Reserved		0h	RO
01 : 00	MV	**Map Value (MV):** The value in the bits below indicate the address range the SATA ports responds to.		00h	RO

Bits	**Mode**	**Primary**		**Secondary**	
		Master		**Master**	
00	Non-combined	Port 0		Port 1	
01	Reserved				
10	Reserved				
11	Reserved				

23.1.5.2 Offset 92h: PCS – Port Control and Status Register

This register is only used in systems that do not support AHCI. In AHCI enabled systems, bits[3:0] must always be set, and the status of the port is controlled through AHCI memory space.

Table 23-33. Offset 92h: PCS – Port Control and Status Register

Description:						
View: PCI		BAR: Configuration		Bus:Device:Function: 0:31:2	Offset Start: 92h Offset End: 92h	
Size: 16 bit		Default: 00h			Power Well: Core	
Bit Range		**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
	15	Reserved	Reserved		0h	RW
14	: 13	Reserved	Reserved		0h	RO
	12	ORM	**OOB Retry Mode (ORM):** When cleared, the SATA controller will not retry after an OOB failure. When set, the SATA controller will continue to retry after an OOB failure until successful (infinite retry)		0h	RW
11	: 08	Reserved	Reserved		0h	RWC
	07	Reserved	**Reserved.**		0h	RO
	06	Reserved	**Reserved.**		0h	RO
	05	P1P	**Port 1 Present (P1P):** Same as P0P, except for port 1.		0h	RO
	04	P0P	**Port 0 Present (P0P):** When set, the SATA controller has detected the presence of a device on port 0. It may change at any time. This bit is cleared when the port is disabled via P0E. This bit is not cleared upon surprise removal of a device.		0h	RO
	03	Reserved	**Reserved.**		0h	RO
	02	Reserved	**Reserved.**		0h	RO
	01	P1E	**Port 1 Enabled (P1E):** When set, the port is enabled. When cleared, the port is disabled. When enabled, the port can transition between the on, partial, and slumber states and can detect devices. When disabled, the port is in the 'off' state and cannot detect any devices.This bit takes precedence over P1CMD.SUD.		0h	RW
	00	POE	**Port 0 Enabled (P0E):**When set, the port is enabled. When cleared, the port is disabled. When enabled, the port can transition between the on, partial, and slumber states and can detect devices. When disabled, the port is in the 'off' state and cannot detect any devices. This bit takes precedence over P0CMD.SUD.		0h	RW

23.1.6 Serial ATA Capability Registers

Start	End	Symbol	Name
A8	AB	SATACR0	Serial ATA Capability Register 0
AC	AF	SATACR1	Serial ATA Capability Register 1

23.1.6.1 Offset A8h: SATACR0 – Serial ATA Capability Register 0

This register shall be read-only 0 when CC.SCC is 01h.

Table 23-34. Offset A8h: SATACR0 – Serial ATA Capability Register 0

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2		**Offset Start:** A8h **Offset End:** ABh	
Size: 32 bit	**Default:** 00100012h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved			00h	RO
23 : 20	MAJREV	**Major Revision (MAJREV):** Major revision number of the SATA Capability Pointer implemented.			1h	RO
19 : 16	MINREV	**Minor Revision (MINREV):** Minor revision number of the SATA Capability Pointer implemented.			0h	RO
15 : 08	NEXT	**Next Capability Pointer (NEXT):** Points to the next capability structure. 00h indicates this is the last capability pointer.			00h	RO
07 : 00	CAP	**Capability ID (CAP):** This value of 12h has been assigned by the PCI SIG to designate the SATA Capability Structure.			12h	RO

23.1.6.2 Offset ACh: SATACR1 – Serial ATA Capability Register 1

This register shall be read-only 0 when CC.SCC is 01h.

Table 23-35. Offset ACh: SATACR1 – Serial ATA Capability Register 1

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2		**Offset Start:** ACh **Offset End:** AFh	
Size: 32 bit	**Default:** 00000048h				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved			00h	RO

Table 23-35. Offset ACh: SATACR1 – Serial ATA Capability Register 1

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** ACh **Offset End:** AFh	
Size: 32 bit	**Default:** 00000048h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	BAROFST	**BAR Offset (BAROFST):** Indicates the offset into the BAR where the Index/Data pair are located (in Dword granularity). The Index and Data I/O registers are located at offset 10h within the I/O space defined by LBAR. A value of 004h indicates offset 10h. 000h = 0h offset 001h = 4h offset 002h = 8h offset 003h = Bh offset 004h = 10h offset FFFh = 3FFFh offset (max 16KB)		004h	RO
03 : 00	BARLOC	**BAR Location (BARLOC):** Indicates the absolute PCI Configuration Register address of the BAR containing the Index/Data pair (in Dword granularity). The Index and Data I/O registers reside within the space defined by LBAR in the SATA controller. A value of 8h indicates offset 20h, which is LBAR. 0000 – 0011b = reserved 0100b = 10h => BAR0 0101b = 14h => BAR1 0110b = 18h => BAR2 0111b = 1Ch => BAR3 1000b = 20h => LBAR 1001b = 24h => BAR5 1010 – 1110b = reserved 1111b = reserved		8h	RO

23.1.7 Additional Configuration Registers

23.1.7.1 Offset C0h: ATC – APM Trapping Control Register

Table 23-36. Offset C0h: ATC – APM Trapping Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** C0h **Offset End:** C0h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	RO
03	SST	**Secondary Slave Trap (SST):** Enables trapping and SMI# assertion on legacy I/O accesses to 170h-177h and 376h. The active device on the secondary interface must be device 1 for the trap and/or SMI# to occur.		0h	RW

Table 23-36. Offset C0h: ATC – APM Trapping Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** C0h **Offset End:** C0h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	SPT	**Secondary Master Trap (SPT):** Enables trapping and SMI# assertion on legacy I/O accesses to 170h-177h and 376h. The active device on the secondary interface must be device 0 for the trap and/or SMI# to occur.		0h	RW
01	PST	**Primary Slave Trap (PST):** Enables trapping and SMI# assertion on legacy I/O accesses to 1F0h-1F7h and 3F6h. The active device on the primary interface must be device 1 for the trap and/or SMI# to occur.		0h	RW
00	PMT	**Primary Master Trap (PMT):** Enables trapping and SMI# assertion on legacy I/O accesses to 1F0h-1F7h and 3F6h. The active device on the primary interface must be device 0 for the trap and/or SMI# to occur.		0h	RW

23.1.7.2 Offset C4h: ATS – ATM Trapping Status Register

Table 23-37. Offset C4h: ATS – ATM Trapping Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** C4h **Offset End:** C4h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	RO
03	SST	**Secondary Slave Trap (SST):** Indicates that a trap occurred to the secondary slave device.		0h	RWC
02	SPT	**Secondary Master Trap (SPT):** Indicates that a trap occurred to the secondary master device.		0h	RWC
01	PST	**Primary Slave Trap (PST):** Indicates that a trap occurred to the primary slave device.		0h	RWC
00	PMT	**Primary Master Trap (PMT):** Indicates that a trap occurred to the primary master device.		0h	RWC

23.1.7.3 Offset D0h: SP – Scratch Pad Register

Table 23-38. Offset D0h: SP – Scratch Pad Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** D0h **Offset End:** D3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DT	**Data (DT):** This is a read/write register that is available for software to use. No hardware action is taken on this register.		0h	RW

23.1.7.4 Offset E0h: BFCS – BIST FIS Control/Status Register

Table 23-39. Offset E0h: BFCS – BIST FIS Control/Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** E0h **Offset End:** E3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 14	Reserved	Reserved		0h	RO
13	Reserved	**Reserved**		0h	RO
12	Reserved	**Reserved**		0h	RO
11	BFS	**BIST FIS Successful (BFS):** This bit is set any time a BIST FIS transmitted by the SATA controller receives an R_OK completion status from the device.		0h	RWC
10	BFF	**BIST FIS Failed (BFF):** This bit is set any time that a BIST FIS transmitted by the SATA controller receives an R_ERR completion status from the device.		0h	RWC
09	P1BFI	**Port 1 BIST FIS Initiate (P1BFI):** When a rising edge is detected on this bit, the SATA controller will initiate a BIST FIS to the device on port 1, using the parameters specified in this register and BFTD1 and BFTD2. The BIST FIS will only be initiated if a device is present and not in the partial or slumber states. After a BIST FIS is successfully completed, software must disable and re-enable PCS.P1E prior to attempting additional BIST FISes or to return the SATA controller to a normal operational mode. If the BIST FIS fails, as indicated by BFF in this register, software can clear then set this bit to initiate another BIST FIS.		0h	RW

Table 23-39. Offset E0h: BFCS – BIST FIS Control/Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** E0h **Offset End:** E3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	P0BFI	**Port 0 BIST FIS Initiate (P0BFI):** When a rising edge is detected on this bit, the SATA controller will initiate a BIST FIS to the device on port 0, using the parameters specified in this register and BFTD1 and BFTD2. The BIST FIS will only be initiated if a device is present and not in the partial or slumber states. After a BIST FIS is successfully completed, software must disable and re-enable PCS.P0E prior to attempting additional BIST FISes or to return the SATA controller to a normal operational mode. If the BIST FIS fails, as indicated by BFF in this register, software can clear then set this bit to initiate another BIST FIS.		0h	RW
07 : 02	BFP	**BIST FIS Parameters (BFP):** These bits form the contents of the upper 6 bits of the BIST FIS Pattern Definition in the BIST FIS transmitted by the SATA controller. This field is not port specific – its contents will be used for any BIST FIS initiated on the SATA controller. The specific bit definitions are: **Bit** / **Symbol** / **Description** 7 / T / Far End Transmit mode 6 / A / Align Bypass mode 5 / S / Bypass Scrambling 4 / L / Far End Retimed Loopback 3 / F / Far End Analog Loopback 2 / P / Primitive bit for use with Transmit mode		00h	RW
01 : 00	Reserved	Reserved.		00h	RO

23.1.7.5 Offset E4h: BFTD1 – BIST FIS Transmit Data 1 Register

Table 23-40. Offset E4h: BFTD1 – BIST FIS Transmit Data 1 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** E4h **Offset End:** E7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DATA	**Data (DATA):** The data programmed into this register will form the contents of the second DW of any BIST FIS initiated by the SATA controller. This register is not port specific – its contents will be used for BIST FIS initiated on any port. Although the 2nd and 3rd DWs of the BIST FIS are only meaningful when the "T" bit of the BIST FIS is set to indicate "Far-End Transmit mode", this register's contents will be transmitted as the BIST FIS 2nd DW regardless of whether or not the T bit is indicated in the BFCS register.		00000000h	RW

23.1.7.6 Offset E8h: BFTD2 – BIST FIS Transmit Data 2 Register

Table 23-41. Offset E8h: BFTD2 – BIST FIS Transmit Data 2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** E8h **Offset End:** EBh	
Size: 32 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DATA	**Data (DATA):** The data programmed into this register will form the contents of the third DW of any BIST FIS initiated by the SATA controller. This register is not port specific – its contents will be used for BIST FIS initiated on any port. Although the 2nd and 3rd DWs of the BIST FIS are only meaningful when the "T" bit of the BIST FIS is set to indicate "Far-End Transmit mode", this register's contents will be transmitted as the BIST FIS 3rd DW regardless of whether or not the "T" bit is set in the BFCS register.		0h	RW

23.1.7.7 Offset F8h: MANID – Manufacturing ID Register

Table 23-42. Offset F8h: MANID – Manufacturing ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:2	**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		0h	RO
23 : 16	SID	**Stepping Identifier (SID):** This field is incremented for each stepping of the part. Note that this field can be used by software to differentiate steppings when the Revision ID may not change. ***Note:*** 00h for A0 stepping ***Note:*** 01h for B0 stepping The value read from this register is the same as that read from Device 31, Function 0, Offset F8h.		Variable	RO
15 : 08	MID	**Manufacturer Identifier (MID):** 0Fh = Intel		0Fh	RO
07 : 00	PID	**Process/Dot Identifier (PID):** Indicates the process as 1263h.		90h	RO

23.2 SATA I/O Mapped Registers

All I/O registers are in the core well. When the Index/Data Pair is enabled these registers use 32 bytes of I/O space, allocated via the LBAR register (configuration offset 20h). When the Index/Data Pair is disabled these registers use 16 bytes of I/O space.

Registers 00-0Fh are only used for legacy operation. Software must not use these registers when running AHCI.

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Start	End	Symbol	Name
00	00	PCMD	Primary Command
02	02	PSTS	Primary Status
04	07	PDTP	Primary Data Table Pointer
08	08	SCMD	Secondary Command
0A	0A	SSTS	Secondary Status

0C	0F	SDTP	Secondary Data Table Pointer
10	13	INDEX	AHCI Register Index
14	17	DATA	AHCI Register Data

Table 23-43. Bus 0, Device 31, Function 2: Summary of SATA Controller Configuration Registers Mapped Through LBAR I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PCMD – Primary Command Register" on page 848	00h
02h	02h	"Offset 02h: PSTS – Primary Status Register" on page 849	00h
04h	07h	"Offset 04h: PDTP – Primary Descriptor Table Pointer Register" on page 849	Variable
10h	13h	"Offset 10h: INDEX – AHCI Index Register" on page 850	00000000h
14h	17h	"Offset 14h: DATA – AHCI Data Register" on page 851	Variable

23.2.1 Primary Devices

23.2.1.1 Offset 00h: PCMD – Primary Command Register

Table 23-44. Offset 00h: PCMD – Primary Command Register

Description:					
View: PCI	**Base Address:** LBAR (IO)		**Bus:Device:Function:** 0:31:2	**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	RO
03	RWC	**Read / Write Control (RWC):** Sets the direction of the bus master transfer: 0 = memory to device, 1 = device to memory. This bit must not be changed when the bus master function is active.		0h	RW
02 : 01	Reserved	Reserved		0h	RO
00	START	**Start/Stop Bus Master (START):** Setting this bit enables bus master operation of the controller. Bus master operation does not actually start unless the Bus Master Enable bit in PCI configuration space is also set. Clearing it halts bus master operation. All state information is lost when this bit is written to '0'; Master mode operation cannot be stopped and then resumed. If this bit is reset while bus master operation is still active and the device has not yet finished its data transfer, the bus master command is said to be aborted. If this bit is cleared to '0' prior to the DMA data transfer being initiated by the drive in a device to memory data transfer, then the ICH5 will not send DMAT to terminate the data transfer. SW intervention (e.g. sending SRST) is required to reset the interface in this condition. This bit is intended to be cleared by software after the data transfer is completed, as indicated by either the ACT bit being cleared in the status register, or the I bit being set in the status register, or both.		0h	RW

23.2.1.2 Offset 02h: PSTS – Primary Status Register

Table 23-45. Offset 02h: PSTS – Primary Status Register

Description:					
View: PCI	**Base Address:** LBAR (IO)		**Bus:Device:Function:** 0:31:2	**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	PRDIS	**PRD Interrupt Status (PRDIS):** This bit is set when the host controller completes execution of a PRD that has its PRD_INT bit set.		0h	RWC
06	D1DC	**Device 1 DMA Capable (D1DC):** A scratchpad bit set SW to indicate that device 1 of this channel is capable of DMA transfers. This bit has no effect on the hardware.		0h	RW
05	D0DC	**Device 0 DMA Capable (D0DC):** A scratchpad bit set by SW to indicate that device 0 of this channel is capable of DMA transfers. This bit has no effect on the hardware.		0h	RW
04 : 03	Reserved	Reserved		0h	RO
02	I	**Interrupt (I):** This bit is set when a device FIS is received with the 'I' bit has been set provided that software has not disabled interrupt via the nIEN bit of Device Control Register.		0h	RWC
01	ERR	**Error (ERR):** This bit is set when the controller encounters an error during the transfer and must stop the transfer. See section 1.5.2 for the list of errors that set this bit		0h	RWC
00	ACT	**Active (ACT):** Set by the host when the START bit is written to the Command register, and cleared by the host when the last transfer for a region is performed, where EOT for that region is set in the region descriptor, and when the START bit is cleared in the Command register and the controller has returned to an idle condition.		0h	RO

23.2.1.3 Offset 04h: PDTP – Primary Descriptor Table Pointer Register

Table 23-46. Offset 04h: PDTP – Primary Descriptor Table Pointer Register

Description:					
View: PCI	**Base Address:** LBAR (IO)		**Bus:Device:Function:** 0:31:2	**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	DBA	**Descriptor Base Address (DBA):** Corresponds to A[31:2]. This table must not cross a 64K boundary in memory. When read, the current value of the pointer is returned		X	RW
01 : 00	Reserved	Reserved		00h	RO

23.2.2 Secondary Devices

23.2.2.1 Offset 08h: SCMD – Secondary Command Register

Same as that of primary device. See the description of "Offset 00h: PCMD – Primary Command Register" on page 848.

23.2.2.2 Offset 0Ah: SSTS – Secondary Status Register

Same as that of primary device. See the description of "Offset 02h: PSTS – Primary Status Register" on page 849.

23.2.2.3 Offset 0Ch: SDTP – Secondary Descriptor Table Pointer Register

Same as that of primary device. See the description of "Offset 04h: PDTP – Primary Descriptor Table Pointer Register" on page 849.

23.2.3 AHCI Index and Data Registers

23.2.3.1 Offset 10h: INDEX – AHCI Index Register

Table 23-47. Offset 10h: INDEX – AHCI Index Register

Description:					
View: PCI	**BAR:** LBAR (IO)		**Bus:Device:Function:** 0:31:2	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 10	Reserved	Reserved		0h	RO
09 : 02	INDEX	**Index (INDEX):** This Index register is used to select the Dword offset of the Memory Mapped AHCI register to be accessed. A Dword, Word or Byte access is specified by the active byte enables of the I/O access to the Data register.		0h	RW
01 : 00	Reserved	Reserved		0h	RO

23.2.3.2 Offset 14h: DATA – AHCI Data Register

Table 23-48. Offset 14h: DATA – AHCI Data Register

Description:					
View: PCI	**BAR:** LBAR (IO)		**Bus:Device:Function:** 0:31:2	**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DATA	**Data (DATA)**— R/W**:** This Data register is a “window” through which data is read or written to the AHCI memory mapped registers. A read or write to this Data register triggers a corresponding read or write to the memory mapped register pointed to by the Index register. The Index register must be setup prior to the read or write to this Data register. Note that a physical register is not actually implemented as the data is actually stored in the memory mapped registers. Since this is not a physical register, the “default” value is the same as the default value of the register pointed to by Index.		Variable	RW

23.3 SATA Memory Mapped Registers

The memory mapped registers within the SATA controller exist in non-cacheable memory space. Additionally, locked accesses are not supported. If software attempts to perform locked transactions to the registers, indeterminate results may occur. Register accesses shall have a maximum size of 64 bits. 64 bit accesses must not cross an 8 byte alignment boundary.

The registers are broken into 2 sections – global control registers and port control registers. All registers that start below address 100h are global and meant to apply to the entire HBA. The port control registers are the same for all ports, and there are as many registers banks as there are ports.

All registers not defined and all reserved bits within registers return ‘0’ when read.

Start	End	Description
00	1F	Generic Host Control
20	9F	Reserved
A0	FF	Vendor Specific registers
100	17F	Port 0 port control registers
180	1FF	Port 1 port control registers
200	27F	Reserved
280	2FF	Reserved
300	3FF	Reserved

Table 23-49. Bus 0, Device 31, Function 2: Summary of SATA Controller Configuration Registers Mapped Through ABAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"Offset 00h: HCAP – HBA Capabilities Register" on page 853	Variable
04h	07h	"Offset 04h: GHC – Global HBA Control Register" on page 855	00000000h
08h	0Bh	"Offset 08h: IS – Interrupt Status Register" on page 856	00000000h
0Ch	0Fh	"Offset 0Ch: PI – Ports Implemented Register" on page 856	00000000h
10h	13h	"Offset 10h: VS – AHCI Version Register" on page 857	00010100h
A0h	A3h	"Offset A0h: SGPO -SPGIO Control Register" on page 857	00000000h
100h, 180h	17Fh, 1FFh	"Offset 100h: PxCLB[0-1] – Port [0-1] Command List Base Address Register" on page 858	Variable
104h, 184h	107h, 187h	"Offset 104h: PxCLBU[0-1] – Port [0-1] Command List Base Address Register" on page 858	Variable
108h, 188h	10Bh, 18Bh	"Offset 108h: PxFB[0-1] – Port [0-1] FIS Base Address Register" on page 859	Variable
10Ch, 18Ch	10Fh, 18Fh	"Offset 10Ch: PxFBU[0-1] – Port [0-1] FIS Base Address Upper 32-bits Register" on page 859	Variable
110h, 190h	113h, 193h	"Offset 110h: PxIS[0-1] – Port [0-1] Interrupt Status Register" on page 860	00000000h
114h, 194h	117h, 197h	"Offset 114h: PxIE[0-1] – Port [0-1] Interrupt Enable Register" on page 861	00000000h
118h, 198h	11Bh, 19Bh	"Offset 118h: PxCMD[0-1] – Port [0-1] Command Register" on page 863	Variable
120h, 1A0h	123h, 1A3h	"Offset 120h: PxTFD[0-1] – Port [0-1] Task File Data Register" on page 866	0000007Fh
124h, 1A4h	127h, 1A7h	"Offset 124h: PxSIG[0-1] – Port [0-1] Signature Register" on page 867	FFFFFFFFh
128h, 1A8h	12Bh, 1ABh	"Offset 128h: PxSSTS[0-1] – Port [0-1] Serial ATA Status Register" on page 868	Variable
12Ch, 1ACh	12Fh, 1AFh	"Offset 12Ch: PxSCTL[0-1] – Port [0-1] Serial ATA Control Register" on page 869	00000000h
130h, 1B0h	133h, 1B3h	"Offset 130h: PxSERR[0-1] – Port [0-1] Serial ATA Error Register" on page 870	00000000h
134h, 1B4h	137h, 1B7h	"Offset 134h: PxSACT[0-1] – Port [0-1] Serial ATA Active Register" on page 872	00000000h
138h, 1B8h	13Bh, 1BBh	"Offset 138h: PxCI[0-1] – Port [0-1] Command Issue Register" on page 872	00000000h
13Ch, 1BCh	13Fh, 1BFh	"Offset 13Ch: PxSNTF[0-1] – Port [0-1] SNotification Register" on page 873	00000000h

23.3.1 Generic Host Controller

Start	End	Symbol	Description
00	03	HCAP	Host Capabilities
04	07	GHC	Global Host Control
08	0B	IS	Interrupt Status
0C	0F	PI	Ports Implemented
10	13	VS	Version

23.3.1.1 Offset 00h: HCAP – HBA Capabilities Register

This register indicates basic capabilities of the HBA to driver software. The RWO bits in this register are only cleared upon PLTRST#.

Table 23-50. Offset 00h: HCAP – HBA Capabilities Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	S64A	**Supports 64-bit Addressing (S64A):** Indicates the S-ATA controller can access 64-bit data structures. The 32-bit upper bits of the port DMA Descriptor, the PRD Base, and each PRD entry are read/write.		1h	RO
30	SCQA	**Supports Native Command Queuing Acceleration (SCQA):** Indicates the SATA controller supports Serial-ATA native command queuing. The HBA will handle DMA Setup FISes natively and will handle the auto-activate optimization through the FIS.		1h	RWO
29	SSNTF	**Supports SNotifiation Register (SSNTF):** The SATA controller supports the SNotification register.		1h	RWO
28	SIS	**Supports Interlock Switch (SIS):** Indicates whether the S-ATA controller supports interlock switches on its ports for use in hot plug operations. This value is loaded by platform BIOS prior to OS initialization. If this bit is set, BIOS must also map the SATAGP pins to the S-ATA controller through GPIO space.		1h	RWO
27	SSS	**Supports Staggered Spin-up (SSS):** Indicates whether the S-ATA controller supports staggered spin-up on its ports, for use in balancing power spikes. This value is loaded by platform BIOS prior to OS initialization.		1h	RWO
26	SALP	**Supports Aggressive Link Power Management (SALP):** Indicates the S-ATA controller supports auto-generating link requests to the partial or slumber states when there are no commands to process.		1h	RWO
25	SAL	**Supports Activity LED (SAL):** Indicates the S-ATA controller supports a single output pin (SATALED#) which indicates activity.		1h	RO
24	SCLO	**Supports Command List Override (SCLO):** When set to '1', indicates that the HBA supports the PxCMD.CLO bit and it's associated function. When cleared to '0', The HBA is not capable of clearing the BSY and DRQ bits in the Status register in order to issue a software reset if these bits are still set from a previous operation.		1h	RWO
23 : 20	ISS	**Interface Speed Support (ISS):** Indicates the maximum speed the S-ATA controller can support is 1.5 Gbps and 3 Gbps on its ports. Speed can be limited in each port by programming PxSCTL.DET to a lower value.		2h	RWO
19	SNZO	**Supports Non-Zero DMA Offsets (SNZO):** Reserved as per AHCI 1.0		0h	RO
18	Reserved	Reserved		0h	RO
17	Reserved	Reserved: BIOS must clear this bit.		1h	RWO
16	Reserved	Reserved		0h	RO
15	PMD	**PIO Multiple DRQ Block (PMD):** The SATA controller support PIO Multiple DRQ Command Block.		1h	RWO

Table 23-50. Offset 00h: HCAP – HBA Capabilities Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14	SSC	**Slumber State Capable (SSC):** The SATA controller supports the slumber state.		1h	RWO
13	PSC	**Partial State Capable (PSC):** The SATA controller supports the partial state.		1h	RWO
12 : 08	NCS	**Number of Command Slots (NCS):** This register field is read only = 1Fh indicating support for 32 slots.		1Fh	RO
07 : 06	Reserved	Reserved		0h	RO
05	SXS	Supports External SATA (SXS): When set to '1', indicates that the HBA has one or more Serial ATA ports that has a signal only connector that is externally accessible. If this bit is set, software may refer to the PxCMD.ESP bit to determine whether a specific port has its signal connector externally accessible as a signal only connector (i.e. power is not part of that connector). when the bit is cleared to '0', indicates that the HBA has no Serial ATA ports that have a signal only connector externally accessible.		0h	RWO
04 : 00	NDS	**Number of Devices (NDS):** Indicates support for 2 devices.		1h	RO

23.3.1.2 Offset 04h: GHC – Global HBA Control Register

This register controls various global actions of the HBA.

Table 23-51. Offset 04h: GHC – Global HBA Control Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	AE	**AHCI Enable (AE):** When set, indicates that an AHCI driver is loaded and communication to the HBA shall be via AHCI mechanisms. This can be used by an HBA that supports both legacy mechanisms (such as SFF-8038i) and AHCI to know when the HBA is running under an AHCI driver. When set, software shall only talk to the HBA using AHCI. The HBA will not have to allow command processing via both AHCI and legacy mechanisms. When cleared, software will only communicate with the HBA using legacy mechanisms. Software shall set this bit to '1' before accessing other AHCI registers.		0h	RW
30 : 02	Reserved	Reserved		0h	RO
01	IE	**Interrupt Enable (IE):** This global bit enables interrupts from the HBA. When cleared (reset default), all interrupt sources from all ports are disabled. When set, interrupts are enabled.		0h	RW
00	HR	**HBA Reset (HR):** When set by SW, this bit causes an internal reset of the HBA. All state machines that relate to data transfers and queuing will return to an idle condition, and all ports will be re-initialized via COMRESET. When the HBA has performed the reset action, it will reset this bit to '0'. A software write of '0' will have no effect. For a description on which bits are reset when this bit is set, see the *AHCI specification*.		0h	RWS

23.3.1.3 Offset 08h: IS – Interrupt Status Register

This register indicates which of the ports within the controller have an interrupt pending and require service.

Table 23-52. Offset 08h: IS – Interrupt Status Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	Reserved	Reserved		0h	RO
01	IPS1	**Interrupt Pending Status Port 1 (IPS1):** If set, indicates that port 2 has an interrupt pending. Software can use this information to determine which ports require service after an interrupt.		0h	RWC
00	IPS0	**Interrupt Pending Status Port 0 (IPS0):** If set, indicates that port 2 has an interrupt pending. Software can use this information to determine which ports require service after an interrupt.		0h	RWC

23.3.1.4 Offset 0Ch: PI – Ports Implemented Register

This register indicates which ports are exposed to the HBA. It is loaded by platform BIOS. It indicates which ports that the device supports are available for software to use. Any available port may not be implemented.

Table 23-53. Offset 0Ch: PI – Ports Implemented Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 0Ch **Offset End:** 0Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	Reserved	Reserved		0h	RO
01	PI1	**Port 1 Implemented (PI1):** If set, the port is available for use. If cleared, the port is not available for use.		0h	RWO
00	PI0	**Port 0 Implemented (PI0):** If set, the port is available for use. If cleared, the port is not available for use.		0h	RWO

23.3.1.5 Offset 10h: VS – AHCI Version Register

This register indicates the major and minor version of the *AHCI specification*. It is BCD encoded. The upper two bytes represent the major version number, and the lower two bytes represent the minor version number. Example: Version 3.12 would be represented as 00030102h. The current version of the specification is 1.1 (00010100h).

Table 23-54. Offset 10h: VS – AHCI Version Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00010100h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	MJR	**Major Version Number (MJR):** Indicates the major version is “1”		0001h	RO
15 : 00	MNR	**Minor Version Number (MNR):** Indicates the minor version is “10”.		0100h	RO

23.3.2 Vendor Specific Registers

This register contains the output bits associated with the two SATA drives and Vendor Specific Message when operating in the AHCI mode. The bits associated with the drives are intended to control LEDs (e.g., activity, fault, and locate LEDs). It is the responsibility of software to write data into this register. Three data bits are allocated per drive.

23.3.2.1 Offset A0h: SGPO - SPGIO Control Register

Table 23-55. Offset A0h: SGPO -SPGIO Control Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	Reserved	**Reserved**”		00000h	RO
11 : 10	Reserved	**Reserved**”		00b	RO
9 :7	D1DAT	Drive 1 Data: Indicates the data to be driven out for drive 1 on SDATAOUT0.		000b	RW
6 :4	D0DAT	Drive 0Data: Indicates the data to be driven out for drive 0 on SDATAOUT0.		000b	RW
03 : 00	SVSM	**SLOAD Vendor Specific message: Indicates the vendor specific message to be driven on to SPGPIO bus on the SLOAD signal**.		0h	RW

23.3.3 Port DMA Registers

23.3.3.1 Offset 100h: PxCLB[0-1] – Port [0-1] Command List Base Address Register

Table 23-56. Offset 100h: PxCLB[0-1] – Port [0-1] Command List Base Address Register

Description:					
View: PCI	BAR: ABAR		Bus:Device:Function: 0:31:2	Offset Start: 100h, 180h Offset End: 17Fh, 1FFh	
Size: 32 bit	Default: Variable			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 10	CLB	**Command List Base Address (CLB):** Indicates the 32-bit base for the command list for this port. This base is used when fetching commands to execute. This address must be 1K aligned as indicated by bits 31:10 being read/write. Note that these bits are not reset on a HBA reset.		Variable	RW
09 : 00	Reserved	Reserved		000h	RO

23.3.3.2 Offset 104h: PxCLBU[0-1] – Port [0-1] Command List Base Address Upper 32-bits Register

Table 23-57. Offset 104h: PxCLBU[0-1] – Port [0-1] Command List Base Address Register

Description:					
View: PCI	BAR: ABAR		Bus:Device:Function: 0:31:2	Offset Start: 104h, 184h Offset End: 107h, 187h	
Size: 32 bit	Default: Variable			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CLBU	**Command List Base Address Upper (CLBU):** Indicates the upper 32-bits for the command list base address for this port. This base is used when fetching commands to execute. Note that these bits are not reset on a HBA reset.		Variable	RW

23.3.3.3 Offset 108h: PxFB[0-1] – Port [0-1] FIS Base Address Register

Table 23-58. Offset 108h: PxFB[0-1] – Port [0-1] FIS Base Address Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 108h, 188h **Offset End:** 10Bh, 18Bh	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 08	FB	**FIS Base Address (FB):** Indicates the 32-bit base for received FISes. This address must be 256-byte aligned as indicated by bits 31:08 being read/write. Note that these bits are not reset on a HBA reset.		XX	RW
07 : 00	Reserved	Reserved		00h	RO

23.3.3.4 Offset 10Ch: PxFBU[0-1] – Port [0-1] FIS Base Address Upper 32-bits Register

Table 23-59. Offset 10Ch: PxFBU[0-1] – Port [0-1] FIS Base Address Upper 32-bits Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 10Ch, 18Ch **Offset End:** 10Fh, 18Fh	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	FBU	**FIS Base Address Upper (FBU):** Indicates the upper 32-bits for the received FIS base for this port. Note that these bits are not reset on a HBA reset.		Variable	RW

23.3.3.5 Offset 110h: PxIS[0-1] – Port [0-1] Interrupt Status Register

Table 23-60. Offset 110h: PxIS[0-1] – Port [0-1] Interrupt Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 110h, 190h **Offset End:** 113h, 193h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	CPDS	**Cold Port Detect Status (CPDS):** The SATA controller does not support cold presence detect.		0h	RO
30	TFES	**Task File Error Status (TFES):** This bit is set whenever the status register is updated by the device and the error bit (bit 0) is set.		0h	RWC
29	HBFS	**Host Bus Fatal Error Status (HBFS):** Indicates that the HBA encountered a host bus error that it cannot recover from, such as a bad software pointer. In PCI, such an indication would be a target or master abort.		0h	RWC
28	HBDS	**Host Bus Data Error Status (HBDS):** Indicates that the HBA encountered a data error (uncorrectable ECC / parity) when reading from or writing to system memory.		0h	RWC
27	IFS	**Interface Fatal Error Status (IFS):** Indicates that the HBA encountered an error on the SATA interface which caused the transfer to stop.		0h	RWC
26	INFS	**Interface Non-fatal Error Status (INFS):** Indicates that the HBA encountered an error on the SATA interface but was able to continue operation.		0h	RWC
25	Reserved	Reserved		0h	RO
24	OFS	**Overflow Status (OFS):** Indicates that the HBA received more bytes from a device than was specified in the PRD table for the command.		0h	RWC
23	Reserved	Reserved.		0h	RWC
22	PRCS	**PhyRdy Change Status (PRCS):** When set to one indicates the internal PhyRdy signal changed state. This bit reflects the state of PxSERR.DIAG.N. This bit is RO and is only cleared when PxSERR.DIAG.N is cleared. Note that the internal PhyRdy signal also transitions when the port interface enters PARTIAL or SLUMBER power management states. PARTIAL and SLUMBER must be disabled when Surprise Removal Notification is desired, otherwise the power management state transitions will appear as false insertion and removal events.		0h	RO
21 : 08	Reserved	Reserved		0h	RO
07	DIS	**Device Interlock Status (DIS):** When set, indicates that a platform interlock switch has been opened or closed, which may lead to a change in the connection state of the device. This bit is only valid in systems that support an interlock switch (HCAP.SIS set). For systems that do not support an interlock switch, this bit will always be '0'.		0h	RWC
06	PCS	**Port Connect Change Status (PCS):** 1=Change in *Current Connect Status*. 0=No change in *Current Connect Status*. This bit reflects the state of PxSERR.DIAG.X. This bit is only cleared when PxSERR.DIAG.X is cleared.		0h	RO
05	DPS	**Descriptor Processed (DPS):** A PRD with the 'I' bit set has transferred all of its data.		0h	RWC

Table 23-60. Offset 110h: PxIS[0-1] – Port [0-1] Interrupt Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 110h, 190h **Offset End:** 113h, 193h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	UFS	**Unknown FIS Interrupt (UFS):** When set to '1' indicates that an unknown FIS was received and has been copied into system memory. This bit is cleared to '0' by software clearing the PxSERR.DIAG.F bit to '0'. Note that this bit does not directly reflect the PxSERR.DIAG.F bit. PxSERR.DIAG.F is set immediately when an unknown FIS is detected, whereas this bit is set when the FIS is posted to memory. Software should wait to act on an unknown FIS until this bit is set to '1' or the two bits may become out of sync.		0h	RO
03	SDBS	**Set Device Bits Interrupt (SDBS):** A Set Device Bits FIS has been received with the 'I' bit set and has been copied into system memory.		0h	RWC
02	DSS	**DMA Setup FIS Interrupt (DSS):** A DMA Setup FIS has been received with the 'I' bit set and has been copied into system memory.		0h	RWC
01	PSS	**PIO Setup FIS Interrupt (PSS):** A PIO Setup FIS has been received with the 'I' bit set, it has been copied into system memory, and the data related to that FIS has been transferred. This bit shall be set even if the data transfer resulted in an error.		0h	RWC
00	DHRS	**Device to Host Register FIS Interrupt (DHRS):** A D2H register FIS has been received with the 'I' bit set, and has been copied into system memory.		0h	RWC

23.3.3.6 Offset 114h: PxIE[0-1] – Port [0-1] Interrupt Enable Register

This register enables and disables the reporting of the corresponding interrupt to system software. When a bit is set ('1') and the corresponding interrupt condition is active, then an interrupt is generated. Interrupt sources that are disabled ('0') are still reflected in the status registers. This register is symmetrical with the Port 0 Status register.

Table 23-61. Offset 114h: PxIE[0-1] – Port [0-1] Interrupt Enable Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 114h, 194h **Offset End:** 117h, 197h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	CPDE	**Cold Presence Detect Enable (CPDE):** The SATA controller does not support cold presence detect.		0h	RO
30	TFEE	**Task File Error Enable (TFEE):** When set, GHC.IE is set, and P0S.TFES is set, the HBA shall generate an interrupt.		0h	RW
29	HBFE	**Host Bus Fatal Error Enable (HBFE):** When set, GHC.IE is set, and P0IS.HBFS is set, the HBA shall generate an interrupt.		0h	RW

Table 23-61. Offset 114h: PxIE[0-1] – Port [0-1] Interrupt Enable Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 114h, 194h **Offset End:** 117h, 197h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
28	HBDE	**Host Bus Data Error Enable (HBDE):** when set, GHC.IE is set, and P0IS.HBDS is set, the HBA shall generate an interrupt.		0h	RW
27	IFE	**Interface Fatal Error Enable (IFE):** When set, GHC.IE is set, and P0IS.IFS is set, the HBA shall generate an interrupt.		0h	RW
26	INFE	**Interface Non-fatal Error Enable (INFE):** When set, GHC.IE is set, and P0IS.INFS is set, the HBA shall generate an interrupt.		0h	RW
25	Reserved	Reserved		0h	RO
24	OFE	**Overflow Enable (OFE):** When set, and GHC.IE and P0IS.OFS are set, the HBA shall generate an interrupt.		0h	RW
23	Reserved	Reserved.		0h	RW
22	PRCE	**PhyRdy Change Interrupt Enable (PRCE):** When set, and GHC.IE is set, and PxIS.PRCS is set, the HBA shall generate an interrupt.		0h	RW
21 : 08	Reserved	Reserved		0h	RO
07	DIE	**Device Interlock Enable (DIE):** When set, and P0IS.DIS is set, the HBA shall generate an interrupt. For systems that do not support an interlock switch, this bit shall be a read-only '0'.		0h	RW
06	PCE	**Port Change Interrupt Enable (PCE):** When set, GHC.IE is set, and P0IS.PCS is set, the HBA shall generate an interrupt.		0h	RW
05	DPE	**Descriptor Processed Interrupt Enable (DPE):** When set, GHC.IE is set, and P0IS.DPS is set, the HBA shall generate an interrupt.		0h	RW
04	UFE	**Unknown FIS Interrupt Enable (UFE):** When set, GHC.IE is set, and PxIS.UFS is set to '1', the HBA shall generate an interrupt.		0h	RW
03	SDBE	**Set Device Bits FIS Interrupt Enable (SDBE):** When set, GHC.IE is set, and P0IS.SDBS is set, the HBA shall generate an interrupt.		0h	RW
02	DSE	**DMA Setup FIS Interrupt Enable (DSE):** When set, GHC.IE is set, and P0IS.DSS is set, the HBA shall generate an interrupt.		0h	RW
01	PSE	**PIO Setup FIS Interrupt Enable (PSE):** When set, GHC.IE is set, and P0IS.PSS is set, the HBA shall generate an interrupt.		0h	RW
00	DHRE	**Device to Host Register FIS Interrupt Enable (DHRE):** When set, GHC.IE is set, and P0IS.DHRS is set, the HBA shall generate an interrupt.		0h	RW

23.3.3.7 Offset 118h: PxCMD[0-1] – Port [0-1] Command Register

Table 23-62. Offset 118h: PxCMD[0-1] – Port [0-1] Command Register (Sheet 1 of 3)

Description:						
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2		**Offset Start:** 118h, 198h **Offset End:** 11Bh, 19Bh	
Size: 32 bit	**Default:** Variable				**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	ICC	**Interface Communication Control (ICC):** This is a four bit field which can be used to control reset and power states of the interface. If the Link layer is currently in the L_IDLE state, writes to this field shall cause the HBA to initiate a transition to the interface power management state requested. If the Link layer is not currently in the L_IDLE state, writes to this field shall have no effect. **Value** / **Definition** Fh–7h: Reserved 6h: **Slumber:** This will cause the HBA to request a transition of the interface to the Slumber state. The SATA device may reject the request and the interface will remain in its current state. 5h–3h: Reserved 2h: **Partial:** This will cause the HBA to request a transition of the interface to the Partial state. The SATA device may reject the request and the interface will remain in its current state. 1h: **Active:** This will cause the HBA to request a transition of the interface into the active state. 0h: **No-Op / Idle:** When software reads this value, it indicates the HBA is ready to accept a new interface control command, although the transition to the previously selected state may not yet have occurred. When system software writes a non-reserved value other than No-Op (0h), the HBA will perform the action and update this field back to Idle (0h). If software writes to this field to change the state to a state the link is already in (i.e. interface is in the active state and a request is made to go to the active state), the HBA will take no action and return this field to Idle. If the interface is in a low power state and the software wants to transition to a different low power state, software must first bring the link to active and then initiate the transition to the desired low power state.		0h	RW
27	ASP	**Aggressive Slumber / Partial (ASP):** When set, and the ALPE bit is set, the HBA will aggressively enter the Slumber state when it clears the PxCI register and the PxSACT register is cleared. When cleared, and the ALPE bit is set, the HBA will aggressively enter the Partial state when it clears the PxCI register and the PxSACT register is cleared.			0h	RW
26	ALPE	**Aggressive Link Power Management Enable (ALPE):** When set, the HBA will aggressively enter a lower link power state (Partial or Slumber) based upon the setting of the ASP bit.			0h	RW
25	DLAE	**Drive LED on ATAPI Enable (DLAE):** When set, the HBA will drive the LED pin active for commands regardless of the state of PxCMD.ATAPI. When cleared, the HBA will only drive the LED pin active for commands if PxCMD.ATAPI is set to '0'.			0h	RW

Table 23-62. Offset 118h: PxCMD[0-1] – Port [0-1] Command Register (Sheet 2 of 3)

Description:					
View: PCI	BAR: ABAR		Bus:Device:Function: 0:31:2	Offset Start: 118h, 198h Offset End: 11Bh, 19Bh	
Size: 32 bit	Default: Variable			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
24	ATAPI	**Device is ATAPI (ATAPI):** When set, the connected device is an ATAPI device. This bit is used by the HBA to control whether or not to generate the desktop LED when commands are active.		0h	RW
23 :22	Reserved	Reserved		0h	RO
21	ESP	External SATA Port (ESP): when set to '1', indicates that this port is routed externally and will be used with an external SATA device. When set to '1' HCAP.SXS must also be set to '1'. When cleared ('0'), indicates that this port is not routed externally and supports internal SATA devices only. ESP is mutually exclusive with the HPCP bit in this register		0h	RWO
20	CPD	**Cold Presence Detection (CPD):** The SATA controller does not support cold presence detect.		0h	RO
19	ISP	**Interlock Switch Attached to Port (ISP):** When interlock switches are supported in the platform (HCAP.SIS set), this indicates whether this particular port has an interlock switch attached. This bit can be used by system software to enable such features as aggressive power management, as disconnects can always be detected regardless of PHY state with an interlock switch. When this bit is set, it is expected that HPCP in this register is also set. The HBA takes no action on the state of this bit – it is for system software only. For example, if this bit is cleared, and an interlock switch toggles, the HBA shall still treat it as a proper interlock switch event. Note that this bit is not reset on a HBA reset.		0h	RWO
18	HPCP	**Hot Plug Capable Port (HPCP):** This indicates whether the this port is connected to a device which can be hot plugged. SATA by definition is hot-pluggable, but not all platforms are constructed to allow the device to be removed (it may be screwed into the chassis, for example). This bit can be used by system software to indicate a feature such as “eject device” to the end-user. The HBA takes no action on the state of this bit – it is for system software only. For example, if this bit is cleared, and a hot plug event occurs, the HBA shall still treat it as a proper hot plug event. Note that this bit is not reset on a HBA reset.		0h	RWO
17	Reserved	Reserved.		0h	RO
16	Reserved	Reserved		0h	RO
15	CR	**Command List Running (CR):** When this bit is set it indicates that the command list DMA engine for the port is running.		0h	RO
14	FR	**FIS Receive Running (FR):** When this bit is set it indicates that the FIS Receive DMA engine for the port is running.		0h	RO
13	ISS	**Interlock Switch State (ISS):** For HBAs that support interlock switches (HCAP.SIS=1), this bit indicates the current state of the interlock switch. A ‘0’ indicates the switch is closed, and a ‘1’ indicates the switch is opened. For HBAs that do not support interlock switches (HCAP.SIS=0), this bit reports ‘0’. Software should only use this bit if both HCAP.SIS and PxCMD.ISP are set to ‘1’.		Variable	RO

Table 23-62. Offset 118h: PxCMD[0-1] – Port [0-1] Command Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 118h, 198h **Offset End:** 11Bh, 19Bh	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
12 : 08	CCS	**Current Command Slot (CCS):** Indicates the current command slot the HBA is processing. This field is valid when the PxCMD.ST bit is set, and is constantly updated by the HBA. This field can be updated as soon as the HBA recognizes an active command slot, or at some point soon after when it begins processing the command. When PxCMD.ST transitions from a '1' to a '0', the HBA will reset this field to '0'. After PxCMD.ST transitions from '0' to '1', the highest priority slot to issue from next is command slot 0. After the first command has been issued, the highest priority slot to issue from next is PxCMD.CCS + 1. For example, after the HBA has issued its first command, if PxCMD.CCS = 0h and PxCI is set to 3h, the next command that will be issued is from command slot 1.		0h	RO
07 : 05	Reserved	Reserved		0h	RO
04	FRE	**FIS Receive Enable (FRE):** When set, the HBA may post received FISes into the FIS receive area pointed to by PxFB and PxFBU. When cleared, received FISes are not accepted by the HBA, except for the first D2H register FIS after the initialization sequence. System software must not set this bit until PxFB (PxFBU) have been programmed with a valid pointer to the FIS receive area. If software wishes to move the base, this bit must first be cleared, and software must wait for the PxCMD.FR bit to be cleared. Software must not clear this bit while PxCMD.ST is set to '1'.		0h	RW
03	CLO	**Command List Override (CLO):** Setting this bit to '1' causes PxTFD.STS.BSY and PxTFD.STS.DRQ to be cleared to '0'. This allows a software reset to be transmitted to the device regardless of whether the BSY and DRQ bits are still set in the PxTFD.STS register. The HBA sets this bit to '0' when PxTFD.STS.BSY and PxTFD.STS.DRQ have been cleared to '0'. A write to this register with a value of '0' shall have no effect. This bit shall only be set to '1' immediately prior to setting the PxCMD.ST bit to '1' from a previous value of '0'. Setting this bit to '1' at any other time is not supported and will result in indeterminate behavior.		0h	RW
02	POD	**Power On Device (POD):** The SATA controller does not support cold presence detect.		1h	RO
01	SUD	**Spin-Up Device (SUD):** This bit is read/write and defaults to 0 for HBAs that support staggered spin-up via HCAP.SSS. This bit is read only '1' for HBAs that do not support staggered spin-up. On an edge detect from '0' to '1', the HBA shall start a COMRESET initialization sequence to the device. Clearing this bit causes no action on the interface.		Variable	RW/RO
00	ST	**Start (ST):** When set, the HBA may process the command list. When cleared, the HBA may not process the command list. Whenever this bit is changed from a '0' to a '1', the HBA starts processing the command list at entry '0'. Whenever this bit is changed from a '1' to a '0', the PxCI register is cleared by the HBA upon the HBA putting the controller into an idle state. See section 10.2.1 of the AHCI spec for restrictions on when PxCMD.ST can be set to '1' and cleared to '0'.		0h	RW

23.3.4 Port Interface Registers (One Set Per Port)

These registers implement various functions of the interface with an SATA device, including the SATA superset registers of the *SATA specification*.

Table 23-63. Port Interface Registers for Ports[1:0]

Start	End	Symbol	Description
120	123	P0TFD	Port 0 Task File Data
124	127	P0SIG	Port 0 Signature
128	12B	P0SSTS	Port 0 Serial ATA Status
12C	12F	P0SCTL	Port 0 Serial ATA Control
130	133	P0SERR	Port 0 Serial ATA Error
134	137	P0SACT	Port 0 Device Status
138	13B	P0CI	Port 0 Command Issue
13C	13F	P0SNTF	Port 0 Serial ATA Notification
1A0	1A3	P1TFD	Port 1 Task File Data
1A4	1A7	P1SIG	Port 1 Signature
1A8	1AB	P1SSTS	Port 1 Serial ATA Status
1AC	1AF	P1SCTL	Port 1 Serial ATA Control
1B0	1B3	P1SERR	Port 1 Serial ATA Error
1B4	1B7	P1SACT	Port 1 Device Status
1B8	1BB	P1CI	Port 1 Command Issue
1BC	1BF	P1SNTF	Port 1 Serial ATA Notification

23.3.4.1 Offset 120h: PxTFD[0-1] – Port [0-1] Task File Data Register

This is a 32-bit register that copies specific fields of the task file when FISes are received. The FISes that contain this information are:

- D2H Register FIS
- PIO Setup FIS
- Set Device Bits FIS (BSY and DRQ are not updated with this FIS)

Table 23-64. Offset 120h: PxTFD[0-1] – Port [0-1] Task File Data Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 120h, 1A0h **Offset End:** 123h, 1A3h	
Size: 32 bit	**Default:** 0000007Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO

Table 23-64. Offset 120h: PxTFD[0-1] – Port [0-1] Task File Data Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 120h, 1A0h **Offset End:** 123h, 1A3h	
Size: 32 bit	**Default:** 0000007Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 08	ERR	**Error (ERR):** Contains the latest copy of the task file error register.		0h	RO
07 : 00	STS	**Status (STS):** Contains the latest copy of the task file status register. Fields of note in this register that affect AHCI: **Bit** / **Field** / **Definition** 7 / BSY / Indicates the interface is busy 6:4 / N/A / Not applicable 3 / DRQ / Indicates a data transfer is requested 2:1 / N/A / Not applicable 0 / ERR / Indicates an error during the transfer. Note that the HBA updates all 8 bits of this register from the received FIS, not just the bits noted above.		7Fh	RO

23.3.4.2 Offset 124h: PxSIG[0-1] – Port [0-1] Signature Register

This is a 32-bit register which contains the initial signature of an attached device when the first D2H Register FIS is received from that device. It is updated once after a reset sequence.

Table 23-65. Offset 124h: PxSIG[0-1] – Port [0-1] Signature Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 124h, 1A4h **Offset End:** 127h, 1A7h	
Size: 32 bit	**Default:** FFFFFFFFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	SIG	**Signature (SIG):** Contains the signature received from a device on the first D2H register FIS. The bit order is as follows: **Bit** / **Field** 31:24 / LBA High Register 23:16 / LBA Mid Register 15:08 / LBA Low Register 07:00 / Sector Count Register		FFFFFFFFh	RO

23.3.4.3 Offset 128h: PxSSTS[0-1] – Port [0-1] Serial ATA Status Register

This is a 32-bit register that conveys the current state of the interface and host. The HBA updates it continuously and asynchronously. When the HBA transmits a COMRESET to the device, this register is updated to its reset values.

Table 23-66. Offset 128h: PxSSTS[0-1] – Port [0-1] Serial ATA Status Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 128h, 1A8h **Offset End:** 12Bh, 1ABh	
Size: 32 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	Reserved	Reserved		0h	RO
11 : 08	IPM	**Interface Power Management (IPM):** Indicates the current interface state: 0h =Device not present or communication not established 1h =Interface in active state 2h =Interface in PARTIAL power management state 6h =Interface in SLUMBER power management state All other values reserved. This field reflects the interface power management state for both device and host initiated power management.		0h	RO
07 : 04	SPD	**Current Interface Speed (SPD):** Indicates the negotiated interface communication speed. 0h =Device not present or communication not established 1h =Generation 1 communication rate negotiated 2h =Generation 2 communication rate negotiated All other values reserved		0h	RO
03 : 00	DET	**Device Detection (DET):** Indicates the interface device detection and Phy state. 0h = No device detected and Phy communication not established 1h =Device presence detected but Phy communication not established 3h =Device presence detected and Phy communication established 4h =Phy in offline mode as a result of the interface being disabled or running in a BIST loopback mode All other values reserved. Note that, while the true reset default value of this register is 0h, the value read from this register depends on drive presence and the point in time within the initialization process when the register is read.		Variable	RO

23.3.4.4 Offset 12Ch: PxSCTL[0-1] – Port [0-1] Serial ATA Control Register

This is a 32-bit read-write register by which software controls SATA capabilities. Writes to this register result in an action being taken by the host adapter or interface. Reads from the register return the last value written to it.

Table 23-67. Offset 12Ch: PxSCTL[0-1] – Port [0-1] Serial ATA Control Register

Description:					
View: PCI	**BAR:** ABAR	**Bus:Device:Function:** 0:31:2		**Offset Start:** 12Ch, 1ACh **Offset End:** 12Fh, 1AFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Reserved	Reserved		0h	RO
19 : 16	Reserved	Reserved.		0h	RW
15 : 12	SPM	**Select Power Management (SPM):** This field is not used by AHCI.		0h	RW
11 : 08	IPM	**Interface Power Management Transitions Allowed (IPM):** Indicates which power states the HBA is allowed to transition to. If an interface power management state is not allowed via this register field, the HBA will not initiate that state and the HBA will PMNAK$_P$ any request from the device to enter that state. 0h =No interface restrictions 1h =Transitions to the PARTIAL state disabled 2h =Transitions to the SLUMBER state disabled 3h =Transitions to both PARTIAL and SLUMBER states disabled All other values reserved		0h	RW
07 : 04	SPD	**Speed Allowed (SPD):** Indicates the highest allowable speed of the interface. This speed is limited by the HCAP.ISS field. For example, if HCAP.ISS is limited to Gen1 speeds, only Gen1 speeds will be negotiated, even if a 0h or 2h is programmed in this register. 0h =No speed negotiation restrictions 1h =Limit speed negotiation to Generation 1 communication rate 2h =Limit speed negotiation to Generation 2 communication rate All other values reserved.		0h	RW
03 : 00	DET	**Device Detection Initialization (DET):** Controls the HBA's device detection and interface initialization. 0h =No device detection or initialization action requested 1h =Perform interface communication initialization sequence to establish communication. This is functionally equivalent to a hard reset and results in the interface being reset and communications reinitialized. While this field is 1h, COMRESET is continuously transmitted on the interface. Software should leave the DET field set to 1h for a minimum of 1 millisecond to ensure that a COMRESET is sent on the interface. 4h =Disable the Serial ATA interface and put Phy in offline mode. All other values reserved. This field may only be changed to 1h or 4h when PxCMD.ST is '0'. Changing this field to 1h or 4h while the HBA is running results in undefined behavior.		0h	RW

23.3.4.5 Offset 130h: PxSERR[0-1] – Port [0-1] Serial ATA Error Register

Table 23-68. Offset 130h: PxSERR[0-1] – Port [0-1] Serial ATA Error Register (Sheet 1 of 2)

Description:					
View: PCI	BAR: ABAR	Bus:Device:Function: 0:31:2		Offset Start: 130h, 1B0h Offset End: 133h, 1B3h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	DIAG	**Diagnostics (DIAG)** - Contains diagnostic error information for use by diagnostic software in validating correct operation or isolating failure modes:		0000h	RWC

Bits	Description
31:27	Reserved
26	**Exchanged (X):** When set to '1' this bit indicates a COMINIT signal was received. This bit is reflected in the interrupt register PxIS. PCS.
25	**Unrecognized FIS Type (F):** Indicates that one or more FISs were received by the Transport layer with good CRC, but had a type field that was not recognized/known.
24	**Transport state transition error (T):** Indicates that an error has occurred in the transition from one state to another within the Transport layer since the last time this bit was cleared.
23	**Link Sequence Error (S):** Indicates that one or more Link state machine error conditions was encountered. The Link Layer state machine defines the conditions under which the link layer detects an erroneous transition.
22	**Handshake Error (H):** Indicates that one or more R_ERR handshake response was received in response to frame transmission. Such errors may be the result of a CRC error detected by the recipient, a disparity or 8b/10b decoding error, or other error condition leading to a negative handshake on a transmitted frame.
21	**CRC Error (C):** Indicates that one or more CRC errors occurred with the Link Layer.
20	Disparity Error (D): This field is not used by AHCI.
19	**10B to 8B Decode Error (B):** Indicates that one or more 10B to 8B decoding errors occurred.
18	**Comm Wake (W):** Indicates that a Comm Wake signal was detected by the Phy.
17	**Phy Internal Error (I):** Indicates that the Phy detected some internal error.
16	**PhyRdy Change (N):** When set to 1 this bit indicates that the internal PhyRdy signal changed state since the last time this bit was cleared. In the ICH6, this bit will be set when PhyRdy changes from a 0 -> 1 or a 1 -> 0. The state of this bit is then reflected in the PxIS.PRCS interrupt status bit and an interrupt will be generated if enabled. Software clears this bit by writing a 1 to it.

Table 23-68. Offset 130h: PxSERR[0-1] – Port [0-1] Serial ATA Error Register (Sheet 2 of 2)

<table>
<tr><td colspan="2">Description:</td><td colspan="4"></td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: ABAR</td><td colspan="2">Bus:Device:Function: 0:31:2</td><td>Offset Start: 130h, 1B0h
Offset End: 133h, 1B3h</td></tr>
<tr><td colspan="2">Size: 32 bit</td><td colspan="2">Default: 00000000h</td><td colspan="2"></td><td>Power Well: Core</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td>Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>15 : 00</td><td>ERR</td><td>Error (ERR): The ERR field contains error information for use by host software in determining the appropriate response to the error condition.
If one or more of bits 11:8 of this register are set, the controller will stop the current transfer.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>15:12</td><td>Reserved</td></tr><tr><td>11</td><td>Internal Error (E): The SATA controller failed due to a master or target abort when attempting to access system memory.</td></tr><tr><td>10</td><td>Protocol Error (P): A violation of the Serial ATA protocol was detected.</td></tr><tr><td>9</td><td>Persistent Communication or Data Integrity Error (C): A communication error that was not recovered occurred that is expected to be persistent. Persistent communications errors may arise from faulty interconnect with the device, from a device that has been removed or has failed, or a number of other causes.</td></tr><tr><td>8</td><td>Transient Data Integrity Error (T): A data integrity error occurred that was not recovered by the interface.</td></tr><tr><td>7:2</td><td>Reserved</td></tr><tr><td>1</td><td>Recovered Communications Error (M): Communications between the device and host was temporarily lost but was re-established. This can arise from a device temporarily being removed, from a temporary loss of Phy synchronization, or from other causes and may be derived from the PhyNRdy signal between the Phy and Link layers.</td></tr><tr><td>0</td><td>Recovered Data Integrity Error (I): A data integrity error occurred that was recovered by the interface through a retry operation or other recovery action.</td></tr></table></td><td></td><td>0000h</td><td>RWC</td></tr>
</table>

23.3.4.6 Offset 134h: PxSACT[0-1] – Port [0-1] Serial ATA Active Register

Table 23-69. Offset 134h: PxSACT[0-1] – Port [0-1] Serial ATA Active Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 134h, 1B4h **Offset End:** 137h, 1B7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DS	**Device Status (DS):** System software sets this bit for random queuing operations prior to setting the PxCI.CI bit in the same command slot entry. This field is cleared via the Set Device Bits FIS. This field is also cleared when PxCMD.ST is cleared by software. Note that this field is not cleared by COMRESET or SRST.		0h	RWS

23.3.4.7 Offset 138h: PxCI[0-1] – Port [0-1] Command Issue Register

Table 23-70. Offset 138h: PxCI[0-1] – Port [0-1] Command Issue Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 138h, 1B8h **Offset End:** 13Bh, 1BBh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CI	**Commands Issued (CI):** This field is set by software to indicate to the HBA that a command has been built in system memory for a command slot and may be sent to the device. When the HBA receives a FIS which clears the BSY, ERR, and DRQ bits for the command, it clears the corresponding bit in this register for that command slot. This field is also cleared when PxCMD.ST is written from a '1' to a '0' by software.		0h	RWS

23.3.4.8 Offset 13Ch: PxSNTF[0-1] – Port [0-1] SNotification Register

Table 23-71. Offset 13Ch: PxSNTF[0-1] – Port [0-1] SNotification Register

Description:					
View: PCI	**BAR:** ABAR		**Bus:Device:Function:** 0:31:2	**Offset Start:** 13Ch, 1BCh **Offset End:** 13Fh, 1BFh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	RO
15 : 00	PMN_15_0	**PM Notify (PMN[15:0]):** This field indicates whether a particular device with the corresponding PM Port number issued a Set Device Bits FIS to the host with the Notification bit set. PM Port 0h sets bit 0 … PM Port Fh sets bit 15 Individual bits are cleared by software writing 1's to the corresponding bit positions. Note that, while this field is reset to default on a HBA Reset, it is not reset by COMRESET or SRST.		0000h	RWC

23.4 Overview

The SATA host controller contains two modes of operation – a legacy mode using I/O space, and an AHCI mode using memory space. The memory space bit GHC.AE, set by software, indicates to hardware that AHCI is being used. Software must not implement code which mixes the use of legacy mode and AHCI mode.

23.5 Legacy Operation

In this mode of operation, software is performing I/O operations to the controller and SATA devices. The SATA controller is using the shadow registers as described in the *SATA specification*, and performing master/slave operation on the ports. The EP80579 does not support slave operations.

Software must program the DEV bit in the task file as its first operation before programming the rest of the transfer or setting the bus master registers.

23.5.1 Transfer Examples

23.5.1.1 Register FIS Only

If software only wishes to send a command to the SATA device, and does not wish for any ATAPI or data transfer to occur, it will perform PIO operations, setting up the required fields for the command, and either writing to the command register (1F7h for primary, 177h for secondary) or control register (3F6h for primary, 376h for secondary).

Hardware will send the register FIS with the appropriate field set for command or control block, and upon reception of the device-to-host register FIS indicating command completion, will update the shadow block.

23.5.1.2 Non-Queued DMA Data Transfers

The following sections describe a data transfer to and from a device using DMA as the command type. These transfers are all non-queued.

23.5.1.2.1 ATA – Data from Memory to Device

Part I - Software Actions – Command Start

1. Software sets up a PRD table in memory, with one or more entries to accomplish the data transfer of the full command. Software places the address for this table PRD Base register.
2. Software writes to the task file to set up the command, with the final write being to the command register (1F7h, 177h).
3. Software sets the “start” bit in the SATA host controller. This step can be swapped with the previous step, since no action is taken until the data transfer is to begin.

Part II - Hardware Actions – Command Start

Upon seeing the command register written, hardware sends the register FIS to the device, and awaits reception of a PIO setup FIS, device-to-host register FIS, DMA Activate FIS, or DATA FIS.

Part III - Hardware Actions – Data Transfer

1. Since the direction of the transfer is from the host to the device (and the command was a DMA command), hardware will receive a DMA Activate FIS.
2. Upon reception, if the ‘start’ bit has been set, hardware begins fetching data from the locations specified in the PRD table. (Hardware could, and for performance reasons should, have started fetching data prior to seeing the device-to-host register FIS described in Part II).
3. Hardware formulates a DATA FIS and begins transmitting data to the device.
4. Hardware continues fetching PRDs as they become exhausted and fetching data from PRD locations, until the transfer is complete. If the transfer is small enough, this data may fall under a single DATA FIS. The SATA Host controller will send FISes of maximum size to minimize FIS overhead on the data transfer.

Part IV - Hardware Actions – Command Wrap-Up

1. After the last piece of data has been accepted by the device, hardware awaits a device-to-host register FIS.
2. When the register FIS is received, hardware updates its task file shadow block.

Part V - Software Actions – Command Wrap-Up

1. Reading device status and BMIDE status.
2. Complete request to OS.
3. Error handling may occur, including device reset & DMA engine re-initialization.

23.5.1.2.2 ATA – Data from Device to Memory

The ATA Device-to-Memory command is exactly the same as the ATA Memory-to-Device command, except that in Part III, hardware is receiving DATA FISes and writing data to memory, instead of fetching data from memory and sending DATA FISes. The number of DATA FISes used is device specific. Additionally, a DMA Activate FIS will not be received – the device will simply start sending a DATA FIS.

23.5.1.2.3 ATAPI

ATAPI DMA transfers are a combination of a PIO write command for the packet command transfer, followed by a DMA command for the data transfer. See those sections for a detailed explanation.

PIO data transfers are covered below.

23.5.1.2.4 Write to Device

Part I - Software Actions – Command Start

Software writes to the task file to set up the command, with the final write being to the command register (1F7h, 177h).

Part II - Hardware Actions – Command Start

1. Upon seeing the command register written, hardware sends the register FIS to the device, and awaits reception of either a PIO setup FIS or a device-to-host register FIS.
2. Upon reception of the PIO setup FIS (since the command was a PIO command), hardware updates the shadow block with the contents of the FIS and the current status, and holds the E-Status in reserve. If the “I” bit was set, it generates an interrupt.

Part III - Hardware / Software Actions – Data Transfer

Software writes to the data port as 16-bit quantities. Hardware assembles these writes into its FIFO, and formulates a single DATA FIS to transmit the data. If software falls behind the data transmission rate of the interface (very likely), hardware will insert IDLE characters.

Part IV - Hardware Actions – Command Wrap-Up

After the last piece of data has been accepted by the device, hardware updates the shadow block’s status register with the E-Status field.

Part V - Software Actions – Command Wrap-Up

1. Reading device status and BMIDE status
2. Complete request to OS
3. Error handling may occur, including device reset & DMA engine re-initialization.

23.5.1.2.5 Read from Device

The read ATA command is exactly the same as the write ATA command, except that in Part III, hardware is receiving the DATA FIS and writing data to memory, instead of fetching data from memory and sending a DATA FIS.

23.5.1.3 SW Assisted Queued DMA Transfer

In this mode of operation, SW supports queuing, the SATA device supports queuing, but the SATA host controller does not. There are two general flows, one for a device that does not support the DMA setup FIS, and one for a device that does support the DMA setup FIS:

In both cases, SW is doing the work to determine the tag of the transfer. No special hardware action is taken by the SATA host controller.

23.5.2 Error Handling

23.5.2.1 Errors on DMI

Errors on the memory interface will cause the following behavior:

Cycle Type	Address/Cmd Parity Error	Data Parity Error	TAbort	MAbort
I/O, Config Write	Set DPE bit Do not claim cycle	Set DPE bit Claim Cycle, data dropped Return completion success	NA	NA
I/O, Config Read	Set DPE bit Do not claim cycle	NA	NA	NA
Memory Write (to HC)	Not supported	NA	NA	NA
Memory Read (to HC)	NA	NA	NA	NA
Memory Write (from HC)	NA	NA	NA	MAI Not Supp
Memory Read (from HC)	NA	NA	NA	MAI Not Supp
I/O, Cfg Completion (read and write)	NA	NA	NA	MAI Not Supp
Mem. Read Comp. (from HC)	NA	NA	NA	MAI Not Supp
Mem. Read Comp. (to HC)	• Set DPE, • Do not claim cycle • Set Error bit in bus master I/O space, offset 02h, bit 1 • Stop DMA engine Needs a system reset to recover	Set DPE & DPD, Claim cycle • During PRD data transfer, abort DMA operation and set Error bit in bus master I/O space; • During DMA data transfer, propagate the error to the device through crc error without setting Error Status bit.	Set RTA bit. Set Error bit and abort	Set RMA bit. Set Error bit and abort

23.5.2.2 Errors on SATA Interface

There are several errors that can occur on the SATA interface which may interrupt a data transfer.

There are two aspects from the following tables that are important when.

- Errors that occur during DMA data in transfers (device sending data) that will result in data corruption will set the bus master error status bit (bit 1 of I/O offset 01h for primary, bit 1 of I/O offset 05h for secondary), while it may or may not set the bit during other cases. If the SATA device generates R_ERR on DMA data out transfers (host sending data), the bus master error status bit will be set, while it may or may not be set on other transfers.
- Errors that occur in PIO or DMA data in transfers (device generating data) that will result in data corruption will cause the SATA host controller to generate R_ERR on the SATA interface.

If the bus master error bit does get set for PIO transfers or non-data portions, it is acceptable.

Table 1448 breaks out conditions for the above rules. Additionally, other errors that may occur are listed for information purposes.

Table 23-72. Errors During Non-DATA FIS Reception

Error Type	Host Controller Behavior
Received Disparity Error / Illegal Character (K28.3)	Assume character is correct. Reset disparity counter. Do not set bus master error bit. Do not return R_ERR (still check for CRC errors).
Received Disparity Error / Illegal Character (D)	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will retry)
Calculated different CRC than received or malformed FIS received[1]	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will retry)
Phyready dropping unexpectedly	Send ALIGNs and return link FSM to IDLE. Do not set bus master error bit. (Device will retry)
Illegal FIS length for corresponding FIS type[2]	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will retry).

Table 23-73. Errors During PIO Data FIS Reception

Error Type	Host Controller Behavior
Received Disparity Error/ / Illegal character (K28.3)	Assume character is correct. Reset disparity counter. Do not set bus master error bit. Do not return R_ERR (still check for CRC errors).
Received Disparity Error / Illegal Character (D)	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will not retry)
Calculated different CRC than received or malformed FIS received	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will not retry)
Phyready dropping unexpectedly	Send ALIGNs and return link FSM to IDLE. Do not set bus master error bit. (Device will not retry)
Length of PIO Data FIS not matching transfer count in PIO_SETUP FIS	Return R_ERR at end of FIS. Do not set bus master error bit. (Device will not retry).

Table 23-74. Errors During DMA Data FIS Reception

Error Type	Host Controller Behavior
Received Disparity Error / Illegal Character (K28.3)	Assume character is correct. Reset disparity counter. Do not set bus master error bit. Do not return R_ERR (still check for CRC errors).
Received Disparity Error / Illegal Character (D)	Return R_ERR at end of FIS. Set the bus master error bit. (Device will not retry)
Calculated different CRC than received or malformed FIS received or internal buffer overflows (which could be caused by device violating HOLD-HOLDA latency)	Return R_ERR at end of FIS. Set the bus master error bit. (Device will not retry)
Phyready dropping unexpectedly	Send ALIGNs and return link FSM to IDLE. Set the bus master error bit. (Device will not retry)

Table 23-75. Errors during unknown FIS type[3] reception

Error Type	Host Controller Behavior
Unknown FIS Type	The unknown FIS type is itself an error condition, and will result in bus master error bits being set and R_ERR being returned.

Table 23-76. Errors during FIS transmission

Error Type	Host Controller Behavior
Non-data FIS: Received R_ERR (includes link protocol errors during transmission), or phyrdy dropping unexpectedly	Retry the FIS.
Data FIS: Received R_ERR (includes link protocol errors during transmission), or phyrdy dropping unexpectedly	Set the bus master error bit. Do not retry the FIS.

Notes:
1. Malformed FIS = FIS not constructed according to link layer protocols
2. Illegal length for corresponding FIS type; For example the following FIS types and their corresponding lengths:
   D2H Dma_Activate FIS length = 1DW
   D2H Register FIS = 5 DW
   D2H Pio_Setup FIS = 5DW
   D2H Set-Device-Bits = 2DW
   D2H Bist FIS = 3DW
3. Zero-length D2H Data FIS (i.e. Data FIS header immediately followed by CRC), and any D2H Data FIS following a D2H Dma_Activate FIS are treated as unknown FIS types for all purposes.

23.5.3 Hot Plug Operation

Dynamic hot plug (such as surprise removal) is not supported in legacy mode. However, using the PCS register configuration bits, and power management flows, a device can be powered down by software, and the port can then be powered off, allowing removal and insertion of a new device.

23.5.4 48-Bit (“Large”) LBA Operation Requirements

The SATA host controller supports 48-bit LBA through the host-to-device register FIS when accesses are performed via writes to the task file. The SATA host controller will ensure that the correct data is put into the correct byte of the host-to-device FIS as follows: if only one write is performed, the data goes to the location specified; if a second write is performed, the data from the first write is shifted into the “upper” location, and the data from the second write goes to the location specified.

Suppose a sequence of writes occurred to the taskfile as follows:

- 1f2h (Sector Count)- 21h
- 1f2h (Sector Count) - 22h
- 1f3h (Sector Number - 31h
- 1f3h (Sector Number) - 32h
- 1f4h (Cylinder Low) - 41h
- 1f4h (Cylinder Low) - 42h
- 1f5h (Cylinder High) - 51h
- 1f5h (Cylinder High) - 52h

The resulting FIS when the command or control register is written will have the following values:

DW					
0	Features	Command	C	0000000	FIS Type (27h)
1	Dev / Head	Cyl High (52h)	Cyl Low (42h)		Sector Number (32h)
2	Features (exp)	Cyl High Exp (51h)	Cyl Low Exp (41h)		Sector Num Exp (31h)
3	Control	Reserved (0)	Sector Count Exp (21h)		Sector Count (22h)
4	Reserved (0)	Reserved (0)	Reserved (0)		Reserved (0)

The writes do not have to come in any specific order. All that is necessary is that the writes to these registers act as a FIFO – the second write moves data from the first write into a new location.

There are also special considerations when reading from the task file to support 48-bit LBA operation. Software may need to read all 16-bits. Since the registers are only 8-bits wide and act as a FIFO, a bit must be set in the device/control register, which is at offset 3f6h for primary and 376h for secondary (or their native counterparts).

If software clears bit 7 of the control register before performing a read, the last item written will be returned from the FIFO. If software sets bit 7 of the control register before performing a read, the first item written will be returned from the FIFO.

23.5.5 Power Management Operation

23.5.5.1 Introduction

Power management of the SATA Controller and ports will cover operations of the host controller and the SATA wire. This specification does not cover any power management that an SATA device may do internally that is transparent to the interface.

23.5.5.2 Power State Mappings

The *PCI specification* defines power management states for devices, which will be applied to the SATA host controller. They are:

- D0 – working (required).
- D1 – light sleep (not supported).
- D2 – deeper sleep (not supported).
- D3 – very deep sleep (required). This state is split into two sub-states, $D3_{HOT}$ (can respond to PCI configuration accesses) and $D3_{COLD}$ (cannot respond to PCI configuration accesses). These two sub-states are considered the same, where $D3_{HOT}$ has V_{CC}, but $D3_{COLD}$ does not. This is the only state allowed for the host controller when the system is in an S1-S5 state.

SATA devices may also have multiple power states. From parallel ATA, three device states are supported through ACPI. They are:

- D0 – Device is working and instantly available.
- D1 – device enters when it receives a STANDBY IMMEDIATE command. Exit latency from this state is in seconds.
- D3 – from the SATA device's perspective, no different than a D1 state, in that it is entered via the STANDBY IMMEDIATE command. However, an ACPI method is also called which will reset the device and then cut its power through proprietary chipset methods. (In ICH, this included setting the tri-state bits of the interface, a GPIO which reset the device, and a GPIO to cut power to that device.

Each of these device states are subsets of the host controller's D0 state. This is partially because host controllers (as integrated in Intel components) have not supported host-based power management, and also because the device must be put into one of the lower power states before power could be removed from the host.

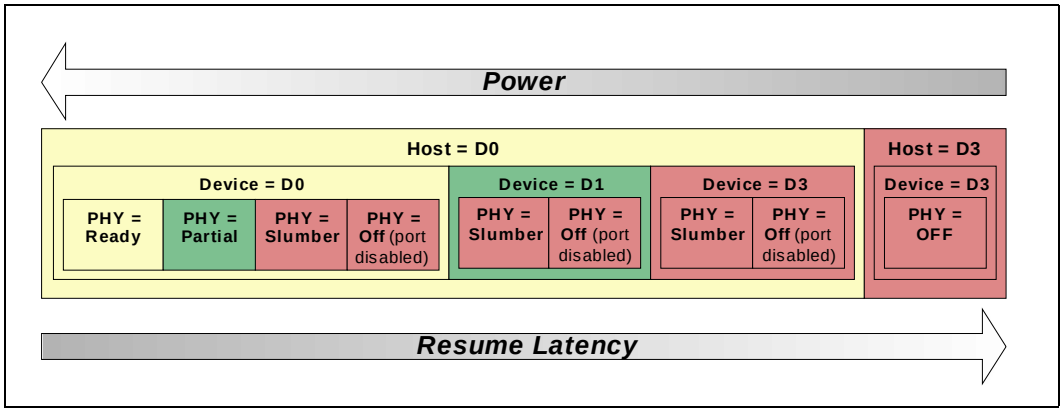
Finally, SATA defines three PHY layer power states, which have no equivalent mappings to parallel ATA. They are:

- PHY READY – PHY logic and PLL are both on and active.
- Partial – PHY logic is powered, but in a reduced state. Exit latency is no longer than 10µs.
- Slumber – PHY logic is powered, but in a reduced state. Exit latency can be up to 10ms.

Since these states have much lower exit latency than the ACPI D1 and D3 states, the SATA Controller defines these states as sub-states of the device D0 state.

Figure 23-1 is an hierarchical view of SATA power states.

Figure 23-1. Legacy Mode Host Controller Power State Hierarchy



23.5.5.3 Power State Transitions

Transitioning between various states is initiated by different levels of software and hardware.

23.5.5.3.1 Partial and Slumber State Entry/Exit

The partial and slumber states are viewed as cheap and easy mechanism to save interface power when the interface is idle. The SATA Controller defines PHY layer power management (as performed via primitives) as a driver operation from the host side, and a device proprietary mechanism on the device side. The SATA Controller will accept device transition types, but will not issue any transitions as a host. All received requests from an SATA device will be ACKed.

When an operation is performed to the SATA Controller such that it needs to use the SATA cable, the controller must check whether the link is in the Partial or Slumber states, and if so, must issue a COM_WAKE to bring the link back online. Similarly, the SATA device must perform the same action.

23.5.5.3.2 Device D1, D3 States

These states are entered after some period of time when software has determined that no commands will be sent to this device for some time. The mechanism for putting a device in these states does not involve any work on the host controller, other then sending commands over the interface to the device. The command most likely to be used in ATA/ATAPI is the “STANDBY IMMEDIATE” command.

23.5.5.3.3 Host Controller D3 state

After the interface and device have been put into a low power state, the host controller may be put into a low power state. This is performed via the PCI power management registers in configuration space.

There are two very important aspects to note when using PCI power management.

1. When the power state is D3, only accesses to configuration space are allowed. Any attempt to access the memory or I/O spaces must result in master abort.
2. When the power state is D3, no interrupts may be generated, even if they are enabled. If an interrupt status bit is pending when the controller transitions to D0, an interrupt may be generated.

When the controller is put into D3, it is assumed that software has properly shut down the device and disabled the ports. Therefore, there is no need to sustain any values on the port wires. The interface will be treated as if no device is present on the cable, and power will be minimized.

When returning from a D3 state, an internal reset will not be performed. When in legacy mode of operation, the SATA controller does not generate PME#. This includes attach events (since the port must be disabled), or interlock switch events (via the SATAGP pins).

23.5.5.4 SMI Trapping (APM)

The ATC register in configuration space contains control for generating SMI# on accesses to the IDE I/O spaces. These bits map to the legacy ranges only (1f0h-1f7h, 3f4h-3f6h, 170h-177h, and 374h-376h). Trapping will not occur on the native IDE ranges defined by PCMDBA, PCTLBA, SCMDBA, SCTLBA, or LBAR. If the SATA controller is in legacy mode and is using these addresses, accesses to one of these ranges with the appropriate bit set will cause the cycle to not be forwarded to the SATA controller, and for SMI# to be generated.

SMI trapping is specifically supported in the following configurations:

- SATA controller in legacy addressing mode (non-combined).

Additionally, an ATS register bit will get set on an access to these ranges if the corresponding bit in the ATC register is set.

23.5.6 Interrupt Architecture

Table 23-77 summarizes interrupt behavior for MSI and wire-modes. In the table “bits” refers to the 4 possible interrupt bits in I/O space, which are: PSTS.PRDIS (offset 02h, bit 7), PSTS.I (offset 02h, bit 2), SSTS.PRDIS (offset 0Ah, bit 7), and SSTS.I (offset 0Ah, bit 2).

Table 23-77. MSI vs. PCI IRQ Actions

Interrupt Register	Wire-Mode Action	MSI Action
All bits ‘0’	Wire inactive	No action
One or more bits set to ‘1’	Wire active	Send message
One or more bits set to ‘1’, new bit gets set to ‘1’	Wire active	Send message
One or more bits set to ‘1’, software clears some (but not all) bits	Wire active	Send message
One or more bits set to ‘1’, software clears all bits	Wire inactive	No action
Software clears one or more bits, and one or more bits is set on the same clock.	Wire active	Send message

23.5.7 Staggered Spin-up

To support staggered spin-up with legacy software, the AHCI memory space register HCAP.SSS must be cleared, and the configuration register PCS is used to enable/disable the port.

23.5.8 HW/SW Operation for Detecting an SATA Device Presence

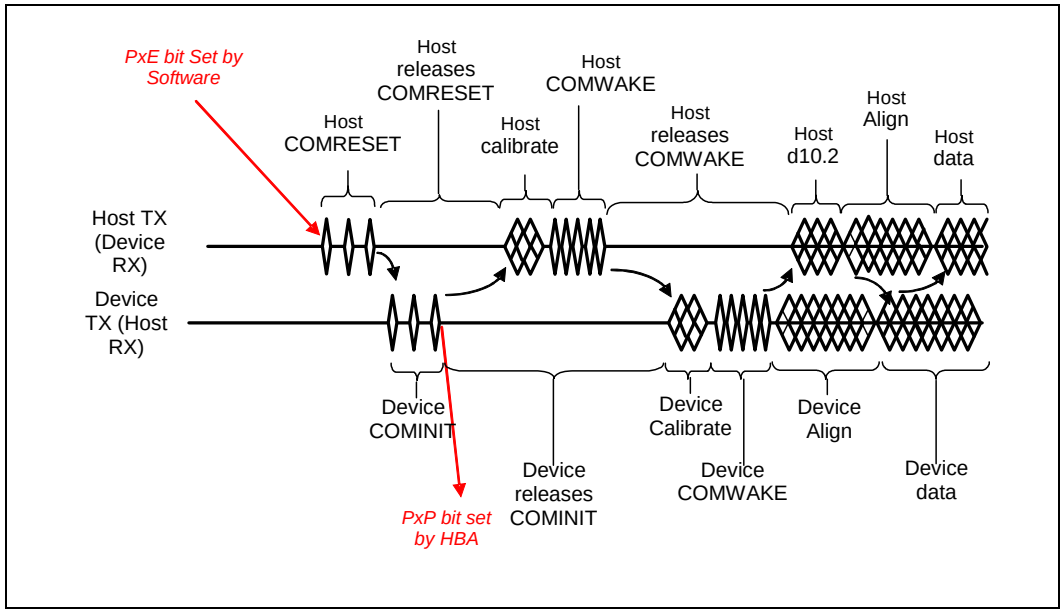
23.5.8.1 Introduction

In legacy mode, the SATA controller does not generate interrupts based on hot plug/unplug events. However, the SATA PHY does know when a device is connected (if not in a partial or slumber state), and it is beneficial to communicate this information to host software as this will greatly reduce boot times and resume times.

23.5.8.2 Hardware Flow

The flow for using these bits is shown in Table 23-2. The ‘PxE’ bit refers to PCS.P0E, and PCS.P1E bits, depending on the port being checked, and the ‘PxP’ bit refers to the PCS.P0P, and PCS.P1P bits, depending on the port being checked.

Figure 23-2. Hardware Flow for Port Enable / Device Present Bits



Notes:
1. The SATA host controller's COMRESET length will be 6 data bursts.

23.5.8.3 Software Flow

The software flow will be as follows:

- Sometime after power-on reset or resume from suspend, software will set the PCS.PxE bits, depending on which ports it wants to scan. They can both be set together.
- The *SATA specification* indicates that the COMRESET sequence to device detection is to be 880 us. For best results, software should wait some period longer than this, such as 10 ms.
- Software will read the PCS.PxP bits. If the bit is set, a device is present. If the bit is cleared, a device is not present.

If a port was disabled, software checks to see if a new device is connected by periodically re-enabling the port, and waiting 2-3ms to see if a device is present. If one is not, it can disable the port and check again later. If a port remains enabled, software can periodically poll PCS.PxP to see if a new device is connected.

23.5.9 SMI Generation

The SATA controller can generate SMIs on behalf of accesses to the task file. These SMIs are used to support legacy operating systems running APM (advanced power management). These SMIs are not used when the controller is an ACPI or PCI power managed device.

To support SMI generation, trap registers exist in power management space. These registers can be found at the PMBASE + 48h region. The register bits are:

D0_TRP_EN: Primary IDE master (port 0)

D1_TRP_EN: Primary IDE slave (port 1)

These bits, when set, affect the legacy IDE ranges 1F0h – 17Fh, 3F6h, 170h – 17Fh, 376h, and the range pointed to by the LBAR register in configuration space. If native IDE is enabled, trapping will not be performed. If AHCI is enabled, trapping will not be performed.

23.5.10 LED

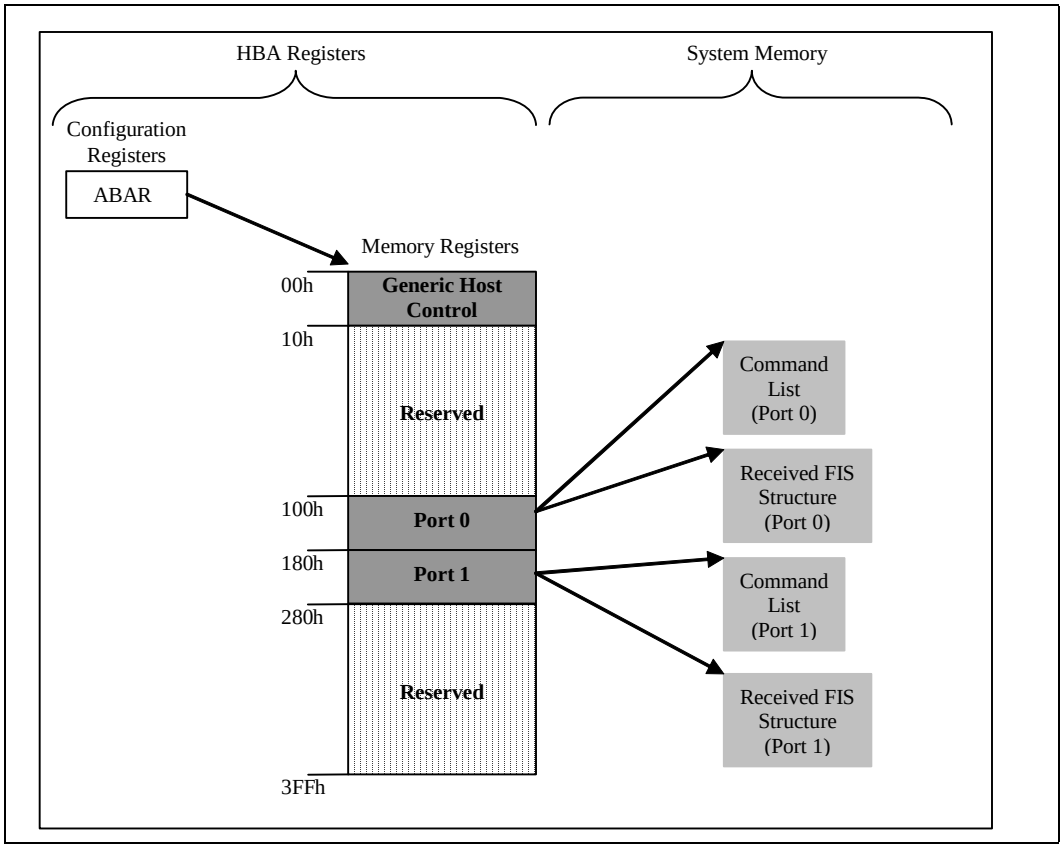
The LED must be driven whenever the BSY bit is set in either port. The LED output will go directly to SATALED#, an active-low open-collector output. When SATALED# is low, the LED is active. When SATALED# is tri-stated, the LED is inactive.

23.6 AHCI Operation

23.6.1 System Memory Structures

The serial ATA controller supports 2ports, and each port supports 32 commands. The command list and received FIS may live in 64-bit space

Figure 23-3. Port System Memory Structure



23.6.2 Error Reporting and Recovery

Error Reporting and Recovery

All errors within an HBA occur within ports. There are no errors that apply to the entire host controller.

23.6.2.1 Error Types

23.6.2.1.1 System Memory Errors

System memory errors such as target abort, master abort, and parity may cause the host to stop processing the currently running command. These are serious errors that cannot be recovered from without software intervention.

A master/target abort error occurs when system software has given a pointer to the HBA that does not exist in physical memory. When this occurs, the HBA aborts the transfer (if necessary). When this is complete, the HBA sets PxIS.HBFS. If PxIE.HBFE and GHC.IE are set, the HBA shall also generate an interrupt.

A data error (such as CRC or parity), may or may not be transient. If the error occurred on a fetch of a CFIS, PRD entry or command list, the HBA shall stop. If the error occurred on a data FIS or the ACMD field, the HBA is allowed to stop, but may also continue. When a data error occurs, the HBA aborts the transfer (if necessary). When this is complete, the HBA sets PxIS.HBDS. If PxIE.HBDE and GHC.IE are set, the HBA shall also generate an interrupt.

If the HBA continue after a data error on a data or ACMD field, it shall poison the CRC of the Data FIS it transfers to the device.

23.6.2.1.2 Interface Errors

Interface errors are errors that occur due to electrical issues on the interface, or protocol miscommunication between the device and HBA. Depending on the type of error, different bits in the PxSERR register are set. When these bits are set, either PxIS.IFS (fatal) or PxIS.INFS (non-fatal) shall be set, and if enabled, the HBA shall generate an interrupt.

Conditions that cause PxIS.IFS/PxIS.INFS to be set are:

- In the PxSERR.ERR field, the P bit is set to '1'
- In the PxSERR.DIAG field, the C or H bit is set to '1'
- PhyRdy drops unexpectedly

Examples of these types of errors are below, with the corresponding PxSERR bit that is set if appropriate.

The only difference between PxIS.IFS and PxIS.INFS being set is the type of FIS that is being transmitted/received when the error occurs. If the error occurred during a non-Data FIS, the FIS must be retransmitted, so the error is non-fatal and PxIS.INFS is set. If the error occurred during a Data FIS, the transfer shall stop, so the error is fatal and PxIS.IFS is set.

In the case of a non-Data FIS error, between seeing a non-Data FIS fail and the attempt to re-transmit, the HBA may receive other FISes from the device (this will most likely happen when performing native command queuing commands). When this occurs, the HBA must accept the FIS, perform the correct actions, and then retry the failed FIS.

If the HBA was transmitting a Data FIS it does not retry the FIS and waits for software to clear the PxCMD.ST bit to '0'. The HBA shall retransmit a non-Data FIS continuously after a failure until either the transfer succeeds or system software stops the controller by clearing PxCMD.ST to '0'.

- Received Disparity Error / Illegal Character (K28.3): When a disparity error is encountered, the HBA assumes the disparity of this character is correct and resets the running disparity counter in the 8b/10b decoder. No error bits are set.
- Received Disparity Error / Illegal Character (D): When this occurs, the HBA returns R_ERR at the end of the FIS. It sets PxSERR.DIAG.B. Note that PxIS.IFS/PxIS.INFS shall not be set; the CRC error that is likely to occur due to this event will set the appropriate bit. If there is an illegal character outside the FIS boundaries, the HBA may ignore the event and is not required to set PxSERR.DIAG.B.
- PhyRdy Dropping Unexpectedly: When this occurs, the HBA returns to idle. If the PhyRdy signal dropped during the middle of a command, the HBA may have to be restarted. If the PhyRdy signal dropped outside of a FIS, neither the PxIS.IFS nor the PxIS.INFS bits shall be set.
- Calculated Different CRC than Received: When this occurs, the HBA returns R_ERR and returns to idle. It sets PxSERR.DIAG.C
- Incorrect FIS or FIS with Illegal Length for Corresponding FIS Type Received: When this occurs, the HBA returns R_ERR at end of the FIS, shall not post the FIS to memory, and returns to idle. It sets PxSERR.ERR.P. This can only be done for supported FIS types. An unknown FIS is not considered an illegal FIS, unless the length received is more than 64 bytes. If an unknown FIS arrives with length <= 64 bytes, it is posted and the HBA continues normal operation.
- Internal Buffer Overflow: This occurs when the HBA sends a HOLD, but a HOLDA was not received quickly enough by the HBA, and the HBA's internal data FIFOs overflow. The HBA returns R_ERR at the end of the FIS. It sets PxSERR.ERR.P.
- HBA Receives R_ERR: If the HBA receives an R_ERR to a FIS it was transmitting, it sets PxSERR.DIAG.H.
- FIS received from a device, where both BSY and DRQ are to be updated in the Status register and both PxTFD.STS.BSY and PxTFD.STS.DRQ are cleared: When this occurs, the HBA returns R_OK, does not set any error bits, and does not update any registers or the received FIS area based on the received FIS (i.e. the FIS is ignored).
- It is system software's responsibility to check the PxSERR register periodically to determine if the interface is operating cleanly, and take appropriate actions (such as going down to Generation1 speed if operating at a higher speed or notifying the user) when interface errors occur.

23.6.2.1.3 Device Errors

When a FIS arrives that updates the taskfile, the HBA checks to see if PxTFD.STS.ERR is set. If it is, and PxIE.TFEE is set, the HBA shall generate an interrupt and stop processing any more commands.

23.6.2.1.4 Command List Overflow

Command list overflow is defined as software building a command table that has fewer total bytes than the transaction given to the device. On device writes, the HBA will run out of data, and on reads, there will be no room to put the data.

For an overflow on data read, either PIO or DMA, the HBA shall set PxIS.OFS, and if enabled via PxIE.OFE and GHC.IE, generate an interrupt. When this condition occurs on data reads, the HBA shall make a best effort to continue, however the HBA may not be able to recover without software intervention. Overflow is a serious error, thus software should perform a fatal error recovery procedure to ensure that the HBA is brought back to a known condition before continuing. For an overflow on writes, the HBA may transmit HOLDs to the device since it does not have any data to satisfy the request size; a COMRESET is required by software to clean up from this serious error.

For an overflow on data writes with DMA, the HBA does not know there is more data until it receives the next DMA Activate. When this occurs, it may optionally set PxIS.OFS and attempt to terminate the transfer. However, this is a fatal condition, and an HBA is allowed to hang on the transfer. For PIO writes, the HBA receives the PIO Setup FIS and therefore knows the length, and therefore may optionally set PxIS.OFS. However, by not satisfying the length, the transfer shall end in an error, and software must recover. Therefore setting PxIS.OFS is optional for both DMA and PIO data write conditions. Detecting overflow and setting PxIS.OFS on native command queuing commands is optional.

23.6.2.1.5 Command List Underflow

Command list underflow is defined as software building a command table that has more total bytes than the transaction given to the device.

For data writes, both PIO and DMA, the device shall detect an error and end the transfer. These errors are most likely going to be fatal errors that will cause the port to be restarted. For data reads, the HBA shall update its PRD byte count with the total number of bytes received from the last FIS, and may be able to continue normally, but is not required to.

The HBA is not required to detect underflow conditions for native command queuing commands.

23.6.2.1.6 Native Command Queuing Tag Errors

The HBA does not actively check incoming DMA Setup FISes to ensure that the PxSACT register bit for that slot is set.

The reason for this is if the device gives an incorrect tag, it could just as likely be for a tag that is active. In this case, the HBA would see no error, although the data transfer that occurs is incorrect. Therefore, there is little benefit in the HBA checking for inactive tags. Just as in the wrong active tag case, the data transfer that occurs will be incorrect.

Existing error mechanisms, such as host bus failure, or bad protocol, are used to recover from this case.

23.6.2.1.7 PIO Data Transfer Errors

In accordance with Serial ATA 1.0a, Data FISes prior to the final Data FIS must be an integral number of Dwords. If the HBA receives an intermediate Data FIS transfer request that is not an integral number of Dwords, the HBA shall set PxSERR.ERR.P to '1', set PxIS.IFS to '1' and stop running until software restarts the port.

The HBA shall ensure that the size of the Data FIS received during a PIO command matches the size in the Transfer Count field of the preceding PIO Setup FIS. If the Data FIS size does not match the Transfer Count field in the preceding PIO Setup, the HBA shall respond with R_ERR to the Data FIS, set PxSERR.ERR.P to '1', set PxIS.IFS to '1', and then stop running until software restarts the port.

23.6.2.2 Error Recovery

23.6.2.2.1 HBA Aborting a Transfer

When the HBA detects an error that it cannot recover from, it may need to end the transfer on the SATA interface.

To do this, the HBA asserts SYNC Escape to stop the bad FIS, and when the device is quiescent, returns to idle. The SATA device should send a D2H Register FIS at this point, with the ERR bit set to indicate an error in the transfer.

When aborting a transfer, the HBA does not wait for the D2H Register FIS before proceeding with error recovery (such as setting interrupt status bits and generating interrupts). This is because a device may be in a hung condition and cannot generate the D2H Register FIS.

23.6.2.2.2 Software Error Recovery

When an interrupt is generated due to an error condition, software will attempt to recover. Fatal errors (signified by the setting of PxIS.HBFS, PxIS.HBDS, PxIS.IFS, or PxIS.TFES) will cause the HBA to enter the ERR:Fatal state, and clear PxCMD.CR. In this state, the HBA shall not issue any new commands nor acknowledge DMA Setup FISes to process any native command queuing commands. To recover, the port must be restarted; the port is restarted by clearing PxCMD.ST to '0' and then setting PxCMD.ST to '1'. For non-fatal errors (signified by the setting of PxIS.INFS or PxIS.OFS) the HBA continues to operate.

If the transfer was aborted, the device is expected to send a D2H Register FIS with PxTFD.STS.ERR set to '1' and both PxTFD.STS.BSY and PxTFD.STS.DRQ cleared to '0'. Under this scenario, system software knows that the device is in a stable state and transfers may be restarted without issuing a COMRESET to the device. No FIS will be posted and no register updates will be done based on FISes received after a fatal error has occurred. Received FISes will not be acted on until a COMRESET or a new command is sent to the device, after the error is recovered from appropriately including clearing the PxCMD.ST bit.

For fatal errors, software must determine which commands were not processed and either re-issue them or notify higher level software that the command failed.

To detect an error that requires software recovery actions to be performed, software should check whether any of the following status bits are set on an interrupt: PxIS.HBFS, PxIS.HBDS, PxIS.IFS, and PxIS.TFES. If any of these bits are set, software should perform the appropriate error recovery actions based on whether non-queued commands were being issued or native command queuing commands were being issued.

23.6.3 Hot Plug Operation

If HCAP.SIS is set, the SATA controller uses the SATAGP[3:0] pins as interlock switches.

The EP80579 supports Hot Plug Surprise Removal Notification. However Hot Plug Surprise Removal Notification (without an interlock switch) is **mutually exclusive** with the PARTIAL and SLUMBER power management states.

23.6.4 Power Management Operation

23.6.4.1 Introduction

This section covers power management of the HBA and the Serial ATA interface. This specification does not cover any power management that a Serial ATA device may do internally that is transparent to the interface.

23.6.4.2 Power State Mappings

The *PCI specification* defines power management states for devices, which shall be applied to the HBA. They are:

- D0 - Working (required).
- D1 - Not defined for storage HBAs.
- D2 - Not defined for storage HBAs.
- D3 - Very deep sleep (required). This state is split into two sub-states, D3HOT (can respond to PCI configuration accesses) and D3COLD (cannot respond to PCI configuration accesses). These two sub-states are considered the same, where D3HOT has VCC, but D3COLD does not. PxCMD.ST must be cleared to '0' before entering the D3 power state.

Serial ATA devices may also have multiple power states. Each of these device states are subsets of the HBA's D0 state. They are:

- D0 - Device is working and instantly available.
- D1 - Device enters when it receives a STANDBY IMMEDIATE command. Exit latency from this state is in seconds.
- D2 - Not currently defined for Serial ATA devices.
- D3 - Device enters when it receives a SLEEP command. Exit latency from this state is in seconds.

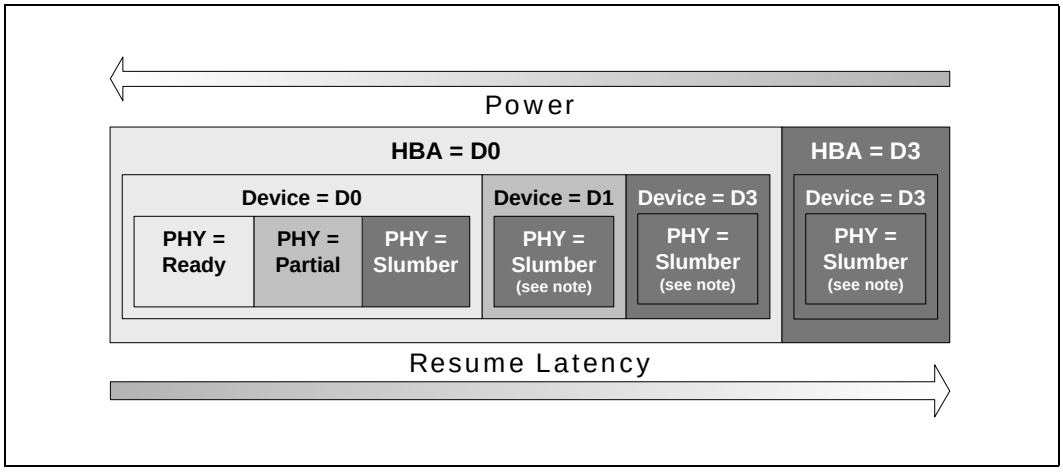
Finally, Serial ATA defines three Phy layer (or interface) power states. They are:

- Phy Ready - Phy logic and PLL are both on and active.
- Partial - Phy logic is powered, but in a reduced state. Exit latency is no longer than 10µs.
- Slumber - Phy logic is powered, but in a reduced state. Exit latency can be up to 10ms.

Since these states have much lower exit latency than the D1 and D3 states, AHCI defines these states as sub-states of the device D0 state.

The following picture gives a hierarchical view of power states of Serial ATA.

Figure 23-4. Power State Hierarchy



Note: The Phy is not required to be in a Slumber state when the device is in a D1 or D3 state, nor is it required to be in a Slumber state when the HBA is in a D3 state. While this may be the likely condition of the interface when the devices connected to the interface are in a low power state, it is not a requirement, and the interface shall break out of these states on a power management event.

23.6.4.3 Power State Transitions

23.6.4.3.1 Interface Power Management

The *Serial ATA 1.0a Specification* defines two lower power interface power management states, Partial and Slumber, in order to save power on the Serial ATA link in power sensitive systems. The Partial and Slumber interface power management states can be initiated by software, the HBA itself, or by the device. The interface power management state is negotiated between the host and the device on the interface using Serial ATA primitives. Any request can be accepted (using the PMACK primitive) or rejected (using PMNACK primitives) based upon current conditions and settings in the device and HBA. The current interface power management state is reflected to software in PxSSTS.IPM.

Device Initiated

By default, a device that supports initiating interface power management states has the capability disabled. To enable this feature, the appropriate SET FEATURES command may be issued to the device. The HBA shall respond to device initiated power management requests as specified by PxSCTL.IPM. A request from the device to enter an interface power management state may be rejected by the HBA if the HBA needs to transmit a FIS to the device.

System Software Initiated

PxCMD.ICC is used by system software to initiate interface power management state transitions. The request to transition to a different interface power management state shall only be acted on by the HBA if the Link layer is currently in the L_IDLE state. If the HBA's Link layer is not in the L_IDLE state when the PxCMD.ICC field is written, the request shall be ignored. The HBA shall not perform a transition directly from Partial to Slumber or from Slumber to Partial based on a new value being written to PxCMD.ICC. If the link is currently in a Partial or Slumber interface power management state, it is software's responsibility to bring the link to the active state before requesting a

transition to a different interface power management state. The time from the request written to PxCMD.ICC until the link is active is bounded by the maximum recovery times from Partial or Slumber as outlined in the *Serial ATA 1.0a Specification*.

HBA Initiated

The HBA may implement aggressive power management, as indicated in HCAP.SALP. Aggressive power management allows the HBA to initiate an interface power management state as soon as there are no commands outstanding to the device. This enables immediate entry into the low power interface state without waiting for software in power sensitive systems. The PxCMD.ALPE bit defines whether the feature is enabled and the PxCMD.ASP field controls whether Partial or Slumber is initiated by the HBA when enabled.

When PxCMD.ALPE is set to '1', if the HBA recognizes that there are no commands to process, the HBA shall initiate a transition to Partial or Slumber interface power management state based upon the setting of PxCMD.ASP. The HBA recognizes no commands to transmit as either:

- PxSACT is set to 0h, and the HBA updates PxCI from a non-zero value to 0h.
- PxCI is set to 0h, and a Set Device Bits FIS is received that updates PxSACT from a non-zero value to 0h.

If the PxSACT and PxCI registers are both cleared to 0h, and the interface is in an active state, the HBA shall not initiate placing the interface into a lower power state, unless PxCMD.ICC is written with an appropriate value.

Before performing a FIS transmission, the HBA must ensure the link is in the active state. If the link is in the Partial or Slumber interface power management state, a COMWAKE must be issued, and the HBA must wait until the link is active before proceeding with transmission of the FIS.

23.6.4.3.2 Software Requirements and Precedence

Software must check HCAP.SSC (Slumber capable) and HCAP.PSC (Partial capable) to determine if the HBA supports interface power management transitions as an initiator or a target. If an interface power management state is not supported, then software shall not write the PxCMD.ICC field nor set the aggressive power management capability to initiate a transition to that state. Software must set the PxSCTL.IPM field to disable transition to any unsupported interface power management state. If HCAP.SSC or HCAP.PSC is cleared to '0', software should disable device-initiated power management by issuing the appropriate SET FEATURES command to the device.

HBA initiated interface power management requests are higher priority than software initiated requests. Thus if the HBA and software request transitions to different states at the same time, the HBA's request shall take precedence over the software request.

23.6.4.3.3 Device D1, D3 States

The D1 and D3 device states are entered when system software has determined that no commands will be sent to the device for some time. To enter these states, software may perform two actions. The first is to issue a command to the device to enter the low power state (STANDY IMMEDIATE for D1, SLEEP for D3), and the second step is to put the interface into a Slumber power management state (by setting PxCMD.ICC to 6h).

Note: It is recommended that the device initiate a Slumber power management state when it receives a command to enter the D1 or D3 state.

23.6.4.3.4 HBA D3 state

After the interface and device have been put into a low power state, the HBA may be put into a low power state. This is performed via the PCI power management registers in configuration space. AHCI only supports the D3 state.

There are two very important aspects to note when using PCI power management.

- When the power state is D3, only accesses to configuration space are allowed. Any attempt to access the register memory space must result in master abort.
- When the power state is D3, no interrupts may be generated, even if they are enabled. If an interrupt status bit is pending when the controller transitions to D0, an interrupt may be generated.

Software must disable interrupts (GHC.IE must be cleared to '0') prior to requesting a transition of the HBA to the D3 state. This precaution by software avoids an interrupt storm if an interrupt occurs during the transition to the D3 state.

PxCMD.ST must be cleared to '0' before entry into the D3 power state.

23.6.4.4 PME

When the HBA is in the D3 state, it may optionally wake based on a change in the device state.

PME must be generated when the HBA is in the D3 state under the following conditions:

- PxIS.PCS is set to '1' due to a native hot plug insertion.
- PxIS.DIS set, indicating an interlock switch has been opened or closed.
- PxIS.CPDS set, indicating cold presence detect state change.
- Set Device Bits FIS received on the interface with the T bit set to ‘1’ and the Notification (‘N’) bit also set to ‘1’.

If any of these bits are set, regardless of the setting of the enables in PxIE and GHC.IE, the HBA shall generate PME#.

For the I^2C approach, the SGPIO Control Register is made visible in the SMBus slave register space. SGPIO control register is mapped to SMBus slave registers at offset 9h, Ah and Bh. The SM Link interface is externally connected to an unused SMBus on the BMC. BMC will read the data, encapsulate it in the IPMI format and drive that data through another SMBus or I^2C bus to the GEM controller. GEM controller then updates the LED drivers with the updated status.

23.7 Additional Information

23.7.1 Mode Switching

Software mode could change from one to another by manipulating the MAP register. There are some specific requirements for the register:

- MAP.SMS and MAP.MV shall only be programmed to values that would result in legal CC.SCC, DID and PI values (See Section 48.1.5.1). If MAP.SMS and MAP.MV are not programmed correctly, hardware shall default the mode back to IDE.
- MAP register must survive across D3hot to D0 power state transitions.
- MAP.SMS shall not be manipulated during runtime operation; i.e. the OS will not do this. However, the BIOS may choose to change it during POST to switch to other software mode.

When switching from one mode to another (via MAP.SMS), the BIOS should ensure that LBAR and ABAR configuration register at offset 20h and 24h are cleared correctly. The BIOS should also ensure that the function disable bits are set or cleared correctly.

Note that when CC.SCC is not 01h, software could also 'change mode' from IDE to AHCI or vice versa by writing to the AE bit of GHC register. This change of software mode during run-time (by using AHCI device driver) is not the kind of mode switching described in this section.

23.7.1.1 AHCI Mode

AHCI mode has the following requirements/characteristics:

For AHCI:

- Default CC.SCC = 06h
- Default PI = 01h
- Has a unique Device ID
- BAR0 to BAR4 (i.e. PCMDBA, PCTLBA, SCMDBA, SCTLBA and LBAR) are functional.
- BAR5 (i.e. ABAR) is functional as a memory BAR.
- MSI is supported.
- MAP.SMS is 01b for AHCI mode.
- MAP.SMS is not applicable when MAP.MV is 10b, indicating combined-mode, in mobile SKUs.
- BIOS should program MAP.MV, then only program MAP.SMS to the desired but legal value.

23.7.1.2 IDE Mode

- IDE mode has the following requirements/characteristics:
- D31F2 shall have CC.SCC = 01h
- D31F2 shall support both native and legacy I/O access, indicating by a PI register default value of 8Ah.
- MSI is not supported.
- BAR0 to BAR4 (i.e. PCMDBA, PCTLBA, SCMDBA, SCTLBA and LBAR) are functional.
- BAR5 is functional as an I/O BAR that implements an index/data pair mechanism for accessing SControl, SError and SStatus registers.
- MAP.SMS must be 00b for IDE mode.
- BIOS should program MAP.MV, then only program MAP.SMS to the desired but legal value.

§ §

24.0 SMBus Controller Functional Description: Bus 0, Device 31, Function 3

24.1 Overview

CMI's IICH contains an SMBus Host interface that allows the processor to communicate with SMBus slaves.

Table 24-1 lists the SMBus signals and the actions taken during various power events

Table 24-1. SMBus signals

Signal Name	Power Plane	During Reset	After Reset	S3	S5	Alt Driver
SMBDATA	Resume	See note		High-Z	High-Z	Peripherals
SMBCLK	Resume			High-Z	High-Z	Peripherals
SMBALERT#	Resume	Can be driven high or low.				Peripherals

Note: SMBDATA and SMBCLK might go active if other devices are using the bus.

Unless specified, all of SMBus logic and registers in this chapter are reset by either CF9 reset or RSMRST#.

24.1.1 Host Controller

The CMI provides a *System Management Bus (SMBus) Specification, Version 2.0*-compliant host controller. The host controller provides a mechanism for the processor to initiate communications with SMB peripherals (slaves). The CMI is also capable of operating in a mode in which it can communicate with I^2C compatible devices.

The CMI can perform SMBus messages with either PEC enabled or disabled. The actual PEC calculation and checking is performed in software. The SMBus Host Controller logic can automatically append the CRC byte if configured to do so.

The CMI SMBus logic exists in Device 31, Function 3 configuration space, and consists of a transmit data path and host controller. The transmit data path provides the data flow logic needed to implement the seven different SMB command protocols and is controlled by the host controller. The logic is clocked by the RTC clock.

The SMBus Address Resolution Protocol (ARP) is supported through software by using the existing host controller commands, except for the new Host Notify command (which is actually a received message).

The programming model of the host controller is combined into two portions: A PCI configuration portion and a system I/O mapped portion. All static configuration, such as the I/O base address, is done via the PCI configuration space. Real-time programming of the host interface is done in system I/O space.

The Host Controller needs to check for parity errors as a target. If it sees an error, it must set the detected parity error bit (bit 15 of status). If bit 6 and bit 8 of the command register are set, it needs to generate SERR#, and set the signalled SERR# bit in the status register (bit 14).

24.1.2 Slave Interface

The slave interface allows an external master to write or read. The write cycles can be used to cause certain events or pass messages, and the read cycles can be used to determine the state of various status bits. The internal Host Controller cannot access the internal Slave Interface.

24.2 SMBus Controller PCI Configuration Register Details

Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values and are read-only. Writes to reserved locations may cause system failure and unpredictable behavior.

Reserved bits are read only.

Table 24-2. Bus 0, Device 31, Function 3: Summary of SMBus Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor ID Register" on page 897	8086h
02h	03h	"Offset 02h: DID: Device ID Register" on page 897	5032h
04h	05h	"Offset 04h: CMD: Command Register" on page 897	0000h
06h	07h	"Offset 06h: DS – Device Status Register" on page 898	0280h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 899	Variable
09h	09h	"Offset 09h: PI: Programming Interface Register" on page 900	00h
0Ah	0Ah	"Offset 0Ah: SCC: Sub Class Code Register" on page 900	05h
0Bh	0Bh	"Offset 0Bh: BCC: Base Class Code Register" on page 900	0Ch
20h	23h	"Offset 20h: SM_BASE: SMB Base Address Register" on page 901	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: SVID Register" on page 901	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem Identification Register" on page 902	0000h
3Ch	3Ch	"Offset 3Ch: INTLN: Interrupt Line Register" on page 902	00h
3Dh	3Dh	"Offset 3Dh: NTPN: Interrupt Pin Register" on page 903	Variable
40h	40h	"Offset 40h: HCFG: Host Configuration Register" on page 903	00h
F8h	FBh	"Offset F8h: MANID: Manufacturer ID Register" on page 904	00010F90h

24.2.1 SMBus Controller PCI Configuration Register Descriptions

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183).

24.2.1.1 Offset 00h: VID: Vendor ID Register

Table 24-3. Offset 00h: VID: Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:**	8086h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	Vendor ID**:** This is a 16-bit value assigned to Intel		8086h	RO

24.2.1.2 Offset 02h: DID: Device ID Register

Table 24-4. Offset 02h: DID: Device ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:**	5032h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device ID:** Indicates the device number assigned by the SIG.		5032h	RO

24.2.1.3 Offset 04h: CMD: Command Register

Table 24-5. Offset 04h: CMD: Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	
10	INTD	**Interrupt Disable:** 0 = Enable (default) 1 = Disables SMBus to assert its PIRQB# signal		0b	RW
09	FBE	**Fast Back to Back Enable:** Reserved as ‘0’.		0b	

Table 24-5. Offset 04h: CMD: Command Register (Sheet 2 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
08	SERR_EN	**SERR# Enable:** 0 = Disables SERR# generation 1 = Enables SERR# generation			0b	RW
07	WCC	**Wait Cycle Control:** Reserved as '0'.			0b	
06	PER	**Parity Error Response:** 0 = Disable 1 = Sets Detected Parity Error bit (D3, F3, 06, bit 15) when a parity error is detected.			0b	RW
05 : 01	Reserved	Reserved			00h	
00	IOSE	**I/O Space Enable:** 0 = Disables access to the SM Bus I/O space registers as defined by the Base Address Register 1 = Enables access to the SM Bus I/O space registers as defined by the Base Address Register			0b	RW

24.2.1.4 Offset 06h: DS – Device Status Register

Table 24-6. Offset 06h: DS – Device Status Register (Sheet 1 of 2)

Description:						
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0280h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** 0 = No parity error detected 1 = Parity error detected			0h	RWC
14	SSE	**Signaled System Error:** 0 = No system error detected 1 = System error detected			0h	RWC
13	RMA	**Received Master Abort:** Reserved as '0'.			0h	
12	RTA	**Received Target Abort:** Reserved as '0'.			0h	
11	STA	**Signaled Target-Abort Status:** 0 = Did not terminate transaction for this function with a target abort 1 = The function is targeted with a transaction that terminates with a target abort			0h	RO
10 : 09	DEVT	**DEVSEL# Timing Status:** This 2-bit field defines the timing for DEVSEL# assertion. These read-only bits indicate the DEVSEL# timing when performing a positive decode. ***Note:*** The CMI generates DEVSEL# with medium time. ***Note:*** It is not clear if a PCI master can write to SMBus controller.			01h	RO

Table 24-6. Offset 06h: DS – Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0280h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	Reserved	Reserved		0h	
07	Reserved	Reserved		1h	
06	Reserved	Reserved		0h	
05	Reserved	Reserved		0h	
04	CAP_LIST	**Capabilities List Indicator:** Hardwired to '0' because there are no capability list structures in this function		0h	RO
03	INTS	**Interrupt Status:** This bit indicates that an interrupt is pending. It is independent from the state of the Interrupt Enable bit in the command register.		0h	RO
02 : 00	Reserved	Reserved		00h	

24.2.1.5 Offset 08h: RID: Revision ID Register

The value reported in this register depends on the value written to the Revision ID in Device 31, Function 0, Offset 08h. See Chapter 19.0, "LPC Interface: Bus 0, Device 31, Function 0," for details.

Table 24-7. Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:**	Variable		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	Indicates the revision identifier. The value reported in this register depends on the value written to the Revision ID in Device 31, Function 0, Offset 08h. This register follows the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

24.2.1.6 Offset 09h: PI: Programming Interface Register

Table 24-8. Offset 09h: PI: Programming Interface Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 09h **Offset End:** 09h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PI	Reserved		00h	RO

24.2.1.7 Offset 0Ah: SCC: Sub Class Code Register

A value of 05h indicates that this device is a SM Bus serial controller.

Table 24-9. Offset 0Ah: SCC: Sub Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:** 05h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCC	Sub Class Code: 05h = IICH SM Bus serial controller		05h	RO

24.2.1.8 Offset 0Bh: BCC: Base Class Code Register

A value of 0Ch indicates that this device is a serial controller.

Table 24-10. Offset 0Bh: BCC: Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:** 0Ch			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BCC	Base Class Code: 0Ch = Serial controller.		0Ch	RO

24.2.1.9 Offset 20h: SM_BASE: SMB Base Address Register

SM_BASE sets the base address in I/O space for the SMBus I/O registers (see Section 24.3, "SMBus Controller I/O-Mapped Configuration Register Details" on page 905). These registers can be mapped anywhere in the I/O space on 32-byte boundaries.

Table 24-11. Offset 20h: SM_BASE: SMB Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 20h **Offset End:** 23h	
Size: 32 bit	**Default:**	00000001h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	
15 : 05	BASE_AD	**Base Address:** Provides the 32 byte system I/O base address for the SMB logic.		0h	RW
04 : 01	Reserved	Reserved		0h	
00	IOSI	**I/O Space Indicator:** This read-only bit always is 1, indicating that the SMB logic is I/O mapped.		1	RO

24.2.1.10 Offset 2Ch: SVID: SVID Register

BIOS sets the value in this register to identify the Subsystem Vendor ID. The SMBus SVID register, in combination with the SMBus Subsystem ID register, enables the operating system to distinguish each subsystem from the others.

Note: The software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.

Table 24-12. Offset 2Ch: SVID: SVID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	Subsystem Vendor ID: The SVID register, in combination with the Subsystem ID (SID) register, enables the operating system (OS) to distinguish subsystems from each other. ***Note:*** Software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.		00h	RWO

24.2.1.11 Offset 2Eh: SID: Subsystem Identification Register

BIOS sets the value in this register to identify the Subsystem ID. The SMBus SID register, in combination with the SMBus SVID register, enables the operating system to distinguish each subsystem from the others.

Note: The software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.

Table 24-13. Offset 2Eh: SID: Subsystem Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	Subsystem ID: The SID register, in combination with the SVID register, enables the operating system (OS) to distinguish subsystems from each other. ***Note:*** Software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.		00h	RWO

24.2.1.12 Offset 3Ch: INTLN: Interrupt Line Register

Table 24-14. Offset 3Ch: INTLN: Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:**	00h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTLN	**Interrupt line:** This data is not used. It is used to communicate to software the interrupt line that the interrupt pin is connected to PIRQB#.		00h	RW

24.2.1.13 Offset 3Dh: NTPN: Interrupt Pin Register

Table 24-15. Offset 3Dh: NTPN: Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:**	Variable		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTPN	**Interrupt Pin:** This reflects the value of D31IP.SMIP in CMI configuration space.		Variable	RO

24.2.1.14 Offset 40h: HCFG: Host Configuration Register

Table 24-16. Offset 40h: HCFG: Host Configuration Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** 40h **Offset End:** 40h	
Size: 8 bit	**Default:**	00h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		00h	
02	I2C_EN	0 = SMBus behavior 1 = Enabled to communicate with I^2C devices. This changes the formatting of some commands.		0h	RW
01	SMB_SMI_EN	0 = SMBus interrupts will not generate an SMI# 1 = Any source of an SMB interrupt is instead be routed to generate an SMI#. Refer to Section 24.7 (Interrupts / SMI#). This bit needs to be set for SMBALERT# to be enabled.		0h	RW
00	HST_EN	0 = Disable the SMBus Host controller 1 = Enable. The SMB Host controller interface is enabled to execute commands. The INTREN bit (offset SM_BASE + 02h, bit 0) needs to be enabled for the SMB Host controller to interrupt or SMI#. The SMB Host controller does not respond to any new requests until all interrupt requests have been cleared.		0h	RW

24.2.1.15 Offset F8h: MANID: Manufacturer ID Register

Table 24-17. Offset F8h: MANID: Manufacturer ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:3	**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:**	00010F90h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved		0h	RO
23 :16	SID	Stepping Identifier: This field increments for each stepping of the part. This field can be used by software to differentiate steppings when the Revision ID may not change. See Section 19.2.8.1, "Offset F8h: MANID: Manufacturer ID Register" on page 757 for cases in which the Revision ID may not increment. ***Note:*** 00h for A0 stepping ***Note:*** 01h for B0 stepping		01h	RO
15 : 08	MID	Manufacturing Identifier: Indicates 0Fh = Intel		0Fh	RO
07 : 00	Reserved	Reserved		90h	

24.3 SMBus Controller I/O-Mapped Configuration Register Details

Note: Warning: Address locations that are not listed are considered reserved register locations. Reserved registers are read only and return all zeros.

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183).

Table 24-18. Bus 0, Device 31, Function 3: Summary of SMBus Controller Configuration Registers Mapped Through SM_BASE I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	“Offset 00h: HSTS: Host Status Register” on page 906	00h
02h	02h	“Offset 02h: HCTL: Host Control Register” on page 908	00h
03h	03h	“Offset 03h: HCMD: Host Command Register” on page 912	00h
04h	04h	“Offset 04h: TSA: Transmit Slave Address Register” on page 912	00h
05h	05h	“Offset 05h: HD0: Data 0 Register” on page 913	00h
06h	06h	“Offset 06h: HD1: Data 1 Register” on page 913	00h
07h	07h	“Offset 07h: HBD: Host Block Data Register” on page 914	00h
08h	08h	“Offset 08h: PEC: Packet Error Check Data Register” on page 915	00h
0Ch	0Ch	“Offset 0Ch: AUXS: Auxiliary Status Register” on page 915	00h
0Dh	0Dh	“Offset 0Dh: AUXC: Auxiliary Control Register” on page 916	00h
0Eh	0Eh	“Offset 0Eh: SMLC: SMLINK_PIN_CTL Register” on page 916	07h
0Fh	0Fh	“Offset 0Fh: SMBC: SMBUS_PIN_CTL Register” on page 917	07h

24.3.1 SMBus Controller I/O-Mapped Configuration Register Descriptions

The SMBus Controller I/O-mapped configuration registers are mapped into I/O space using register SM_BASE (see Section 24.2.1.9, “Offset 20h: SM_BASE: SMB Base Address Register” on page 901).

24.3.1.1 Offset 00h: HSTS: Host Status Register

All status bits are set by hardware and cleared by the software writing a one to the particular bit position. Writing a zero to any bit position has no effect.

Table 24-19. Offset 00h: HSTS: Host Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	BDS	**BYTE_DONE_STS:** 0 = Software can clear this by writing a 1 1 = Host controller received a byte (for Block Read commands) or if it has completed transmission of a byte (for Block Write commands) when the 32-byte buffer is not being used. This bit is set, even on the last byte of the transfer. This bit is not set when transmission is due to the LAN interface heartbeat. This bit has no meaning for block transfers when the 32-byte buffer is enabled. When the last byte of a block message is received, the host controller sets this bit. However, it does not immediately set the INTR bit (bit 1 in this register). When the interrupt handler clears the BYTE_DONE_STS bit, the message is considered complete, and the host controller will then set the INTR bit (and generate another interrupt). Thus, for a block message of n bytes, n+1 interrupts will be generated. The interrupt handler needs to be implemented to handle these cases.		0h	RWC
06	IUS	**In Use Status:** 0 = After a full PCI reset, a read to this bit returns a 0. 1 = After the first read, subsequent reads return a 1. A write of a 1 to this bit resets the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and then own the usage of the host controller. This bit has no other effect on the hardware, and is only used as semaphore among various independent software threads that may need to use the SMBus logic.		0h	RW
05	SMBALERT_STS	**System Bus Alert Status:** 0 = Interrupt or SMI# was not generated by SMBALERT#. Software clears this bit by writing a 1 to it. 1 = The source of the interrupt or SMI# was the SMBALERT# signal. This bit is only cleared by software writing a 1 to the bit position or by CF9 RESET or RSMRST# going low (but not PLTRST#). If the signal is programmed as a GPI, then this bit is never set.		0h	RWC

Table 24-19. Offset 00h: HSTS: Host Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	FAIL	**Failed:** 0 = Software clears this bit by writing a 1 to it. 1 = The source of the interrupt or SMI# was a failed bus transaction. This bit is set in response to the KILL bit being set to terminate the host transaction.		0h	RWC
03	MCERR	**Machine Check Error:** 0 = Software clears this bit by writing a 1 to it. 1 = The source of the interrupt of SMI# was a transaction collision.		0h	RWC
02	DERR	**Device Error:** 0 = Software clears this bit by writing a 1 to it, then deasserts the interrupt or SMI#. 1 = The source of the interrupt or SMI# was due to one of the following: • Illegal Command Field· • Unclaimed Cycle (host initiated) • Host Device Time-out Error • CRC Error		0h	RWC
01	INTR	**Interrupt:** When set, this indicates that the source of the interrupt or SMI# was the successful completion of its last command.		0h	RWC
00	HBSY	**Host Busy:** 0 = Cleared when the current transaction is completed 1 = Indicates that the CMI is running a command from the host interface. No SMB registers must be accessed while this bit is set, except the BLOCK DATA BYTE Register. The BLOCK DATA BYTE Register can be accessed when this bit is set only when the SMB_CMD bits in the Host Control Register are programmed for Block command or I^2C Read command. This is necessary in order to check the DONE_STS bit.		0h	RWC

24.3.1.2 Offset 02h: HCTL: Host Control Register

Note: A read to this register clears the pointer in the 32-byte buffer.

Table 24-20. Offset 02h: HCTL: Host Control Register (Sheet 1 of 4)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	PEC_EN	**Packet Error Check Enable:** 0 = SMBus host controller does not perform the transaction with the PEC phase appended. 1 = Causes the host controller to perform the SMBus transaction with the Packet Error Checking phase appended. For writes, the value of the PEC byte is transferred from the PEC Register. For reads, the PEC byte is loaded in to the PEC Register. This bit must be written prior to the write in which the start bit is set.		0h	RW
06	START	0 = This bit will always return 0 on reads. The HOST_BUSY bit in the Host Status register (offset 00h) can be used to identify when the command is finished. 1 = Writing a 1 to this bit initiates the command described in the SMB_CMD field. All registers should be setup prior to writing a 1 to this bit position.		0h	RW

Table 24-20. Offset 02h: HCTL: Host Control Register (Sheet 2 of 4)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	LAST_BYTE	Used for I²C Read commands as an indication that the next byte is the last one to be received for that block. The algorithm and usage model for this bit is as follows (assume a message of n bytes): 1. When the software sees the BYTE_DONE_STS bit set (bit 7 in the SMBus Host Status Register) for each of bytes 1 through n-2 of the message, the software should then read the Block Data Byte Register to get the byte that was just received. 2. After reading each of bytes 1 to n-2 of the message, the software will then clear the BYTE_DONE_STS bit. 3. After receiving byte n-1 of the message, the software will then set the “LAST BYTE” bit. The software will then clear the BYTE_DONE_STS bit. 4. The CMI then receives the last byte of the message (byte n). However, the state machine sees the LAST BYTE bit set, and instead of sending an ACK after receiving the last byte, it instead sends a NAK. 5. After receiving the last byte (byte n), the software still clears the BYTE_DONE_STS bit. However, the LAST_BYTE bit is irrelevant at that point. ***Notes:*** 1. This bit may be set when the TCO timer causes the SECOND_TO_STS bit to be set. See section Section 18.2.2.6, bit 1 for more details on that bit. The SMBus device driver should clear the LAST_BYTE bit (if it is set) before starting any new command. 2. In addition to I2C Read Commands, the LAST_BYTE bit also causes Block Read/Write cycles to stop prematurely (at the end of the next byte).		0h	RW

Table 24-20. Offset 02h: HCTL: Host Control Register (Sheet 3 of 4)

<table>
<tr><td colspan="6">Description:</td></tr>
<tr><td colspan="2">View: PCI</td><td colspan="2">BAR: SM_BASE (IO)</td><td>Bus:Device:Function: 0:31:3</td><td>Offset Start: 02h
Offset End: 02h</td></tr>
<tr><td>Size: 8 bit</td><td colspan="2">Default: 00h</td><td></td><td colspan="2">Power Well: Resume</td></tr>
<tr><td>Bit Range</td><td>Bit Acronym</td><td colspan="2">Bit Description</td><td>Sticky</td><td>Bit Reset Value</td><td>Bit Access</td></tr>
<tr><td>04 : 02</td><td>SMB_CMD</td><td colspan="2">As shown by the bit encoding below, indicates which command is to be performed. If enabled, the CMI generates an interrupt or SMI# when the command has completed If the value is for a non-supported or reserved command, the CMI will sets the device error (DEV_ERR) status bit and generates an interrupt when the start bit is set. The CMI performs no command, and does not operate until DEV_ERR is cleared.

Bits | Name | Command Description
000 | Quick | The slave address and read/write value (bit 0) are stored in the tx slave address register.
001 | Byte | This command uses the transmit slave address and command registers. Bit 0 of the slave address register determines if this is a read or write command.
010 | Byte Data | This command uses the transmit slave address, command, and DATA0 registers. Bit 0 of the slave address register determines if this is a read or write command. If it is a read, the DATA0 register contains the read data.
011 | Word Data | This command uses the transmit slave address, command, DATA0 and DATA1 registers. Bit 0 of the slave address register determines if this is a read or write command. If it is a read, after the command completes the DATA0 and DATA1 registers contain the read data.
100 | Process Call | This command uses the transmit slave address, command, DATA0 and DATA1 registers. Bit 0 of the slave address register determines if this is a read or write command. After the command completes, the DATA0 and DATA1 registers will contain the read data.
101 | Block | This command uses the transmit slave address, command, and DATA0 registers, and the Block Data Byte register. For block write, the count is stored in the DATA0 register and indicates how many bytes of data will be transferred. For block reads, the count is received and stored in the DATA0 register. Bit 0 of the slave address register selects if this is a read or write command. For writes, data is retrieved from the first n (where n is equal to the specified count) addresses of the SRAM array. For reads, the data is stored in the Block Data Byte register.
110 | I^2C Read | This command uses the transmit slave address, command, DATA0, DATA1 registers, and the Block Data Byte register. The read data is stored in the Block Data Byte register. The CMI will continue reading data until the NAK is received.</td><td></td><td>000h</td><td>RW</td></tr>
</table>

Table 24-20. Offset 02h: HCTL: Host Control Register (Sheet 4 of 4)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04 : 02	SMB_CMD (cont'd)	Bits Name Command Description 111 Block-Process This command uses the transmit slave address, command, DATA0 and the Block Data Byte register. For block write, the count is stored in the DATA0 register and indicates how many bytes of data will be transferred. For block read, the count is received and stored in the DATA0 register. Bit 0 of the slave address register always indicate a write command. For writes, data is retrieved from the first m (where m is equal to the specified count) addresses of the SRAM array. For reads, the data is stored in the Block Data Byte register. ***Note:*** E32B bit in the Auxiliary Control Register must be set for this command to work.		0h	RW
01	KILL	0 = Normal SMBus host controller functionality. 1 = Kills the current host transaction taking place, sets the FAILED status bit, and asserts the interrupt (or SMI#). This bit, once set, must be cleared by software to allow the SMBus host controller to function normally.		0h	RW
00	INTREN	0 = Disable 1 = Enable the generation of an interrupt or SMI# upon the completion of the command		0h	RW

24.3.1.3 Offset 03h: HCMD: Host Command Register

This eight bit field is transmitted by the host controller in the command field of the SMB protocol during the execution of any command.

Table 24-21. Offset 03h: HCMD: Host Command Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 03h **Offset End:** 03h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HST_CMD	This 8-bit field is transmitted by the host controller in the command field of the SMBus protocol during the execution of any command.		00h	RW

24.3.1.4 Offset 04h: TSA: Transmit Slave Address Register

This register is transmitted by the host controller in the slave address field of the SMB protocol. This is the address of the target.

Table 24-22. Offset 04h: TSA: Transmit Slave Address Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 04h **Offset End:** 04h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	ADDRESS	7-bit address of the targeted slave		0000000h	RW
00	RW	Direction of the host transfer. 0 = write 1 = read		0h	RW

24.3.1.5 Offset 05h: HD0: Data 0 Register

Table 24-23. Offset 05h: HD0: Data 0 Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)		**Bus:Device:Function:** 0:31:3	**Offset Start:** 05h **Offset End:** 05h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DATA0_COUNT	This field contains the eight bit data sent in the DATA0 field of the SMB protocol. For block write commands, this register reflects the number of bytes to transfer. This register should be programmed to a value between 1 and 32 for block counts. A count of 0 or a count above 32 will result in unpredictable behavior. The host controller does not check or log illegal block counts.		00h	RW

24.3.1.6 Offset 06h: HD1: Data 1 Register

Table 24-24. Offset 06h: HD1: Data 1 Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)		**Bus:Device:Function:** 0:31:3	**Offset Start:** 06h **Offset End:** 06h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DATA1	This eight bit register is transmitted in the DATA1 field of the SMB protocol during the execution of any command.		00h	RW

24.3.1.7 Offset 07h: HBD: Host Block Data Register

Table 24-25. Offset 07h: HBD: Host Block Data Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 07h **Offset End:** 07h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BDTA	**Block Data:** This is either a register, or a pointer into a 32-byte block array, depending upon whether the E32B bit is set in the Auxiliary Control register. When the E32B bit is cleared, this is a register containing a byte of data to be sent on a block write or read from on a block read. When the E32B bit is set, reads and writes to this register are used to access the 32-byte block data storage array. An internal index pointer is used to address the array, which is reset to 0 by reading the HCTL register (offset 02h). The index pointer then increments automatically upon each access to this register. The transfer of block data into (read) or out of (write) this storage array during an SMBus transaction always starts at index address 0. When the E2B bit is set, for writes, software will write up to 32-bytes to this register as part of the setup for the command. After the Host Controller has sent the Address, Command, and Byte Count fields, it will send the bytes in the SRAM pointed to by this register. When the E2B bit is cleared for writes, software will place a single byte in this register. After the host controller has sent the address, command, and byte count fields, it will send the byte in this register. If there is more data to send, software will write the next series of bytes to the SRAM pointed to by this register and clear the DONE_STS bit. The controller will then send the next byte. During the time between the last byte being transmitted to the next byte being transmitted, the controller will insert wait-states on the interface. When the E2B bit is set for reads, after receiving the byte count into the Data0 register, the first series of data bytes go into the SRAM pointed to by this register. If the byte count has been exhausted or the 32-byte SRAM has been filled, the controller will generate an SMI# or interrupt (depending on configuration) and set the DONE_STS bit. Software will then read the data. During the time between when the last byte is read from the SRAM to when the DONE_STS bit is cleared, the controller will insert wait-states on the interface.This eight bit register is transmitted in the DATA1 field of the SMB protocol during the execution of any command.		00h	RW

24.3.1.8 Offset 08h: PEC: Packet Error Check Data Register

This register contains the 8-bit CRC value that is used as the Packet Error Check on SMBus. For writes, this register is written by software prior to running the command. For reads, this register is read by software after the read command is completed on SMBus.

Table 24-26. Offset 08h: PEC: Packet Error Check Data Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PEC_DATA	This 8-bit register is written with the SMBus PEC data prior to a write transaction. For read transactions, the PEC data is loaded from the SMBus into this register and is then read by software. Software must ensure that the INUSE_STS bit is properly maintained to avoid having this field overwritten by a write transaction following a read transaction.		00h	RW

24.3.1.9 Offset 0Ch: AUXS: Auxiliary Status Register

Table 24-27. Offset 0Ch: AUXS: Auxiliary Status Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		0h	
01	STCO	**SMBus TCO mode:** This is the status bit that reflects the setting of legacy TCO mode vs. Advanced TCO mode. 0 = Indicates that this bit is always zero, since Advanced TCO mode is not supported.		0h	RO
00	CRCE	**CRC Error:** 0 = Software clears this bit by writing a 1 to it. 1 = This bit is set if a received message contained a CRC error. When this bit is set, the DERR bit of the host status register is also set. This bit is set by the controller if a software abort occurs in the middle of the CRC portion of the cycle or an abort happens after the CMI has received the final data bit transmitted by an external slave.		0h	RWC

24.3.1.10 Offset 0Dh: AUXC: Auxiliary Control Register

Table 24-28. Offset 0Dh: AUXC: Auxiliary Control Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		0h	
01	E32B	**Enable 32-byte Buffer:** 0 = The Host Block Data register is a pointer into a single register. 1 = When set, the Host Block Data register is a pointer into a 32-byte buffer. This enables the block commands to transfer or receive up to 32-bytes before the CMI generates an interrupt.		0h	RW
00	AAC	**Automatically Append CRC:** 0 = Does not automatically append the CRC 1 = Automatically appends the CRC This bit must not be changed during SM Bus transactions, or undetermined behavior results. It should be programmed only once during the lifetime of the function.		0h	RW

24.3.1.11 Offset 0Eh: SMLC: SMLINK_PIN_CTL Register

This register is only applicable in the TCO compatible mode.

This register is in the resume well and is reset by CF9 RESET or RSMRST#.

Table 24-29. Offset 0Eh: SMLC: SMLINK_PIN_CTL Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 07h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		0h	
02	SMLINK_CLK_CTL	This read/write bit has a default of 1. 0 = Drives the SMLINK[0] pin low, independent of what the other SMLINK logic would otherwise indicate for the SMLINK[0] pin. 1 = The SMLINK[0] pin is *Not* overdriven low. The other SMLINK logic controls the state of the pin.		1b	RW
01	SMLINK1_CUR_STS	This read-only bit has a default value that is dependent on an external signal level. This pin returns the value on the SMLINK[1] pin. This allows software to read the current state of the pin. 0 = Low 1 = High		1b	RO

Table 24-29. Offset 0Eh: SMLC: SMLINK_PIN_CTL Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 07h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
00	SMLINK0_CUR_STS	This read-only bit has a default value that is dependent on an external signal level. This pin returns the value on the SMLINK[0] pin. 0 = Low 1 = High This allows software to read the current state of the pin.		0h	RO

a. Reset by CF9 RESET or RSMRST#

24.3.1.12 Offset 0Fh: SMBC: SMBUS_PIN_CTL Register

This register is in the resume well and is reset by CF9 RESET or RSMRST#.

Table 24-30. Offset 0Fh: SMBC: SMBUS_PIN_CTL Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Fh **Offset End:** 0Fh	
Size: 8 bit	**Default:** 07h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		00h	RO
02	SMBCLK_CTL	This bit has a default of 1. 0 = Drives the SMBCLK pin low, independent of what the other SMB logic would otherwise indicate for the SMBCLK pin. 1 = The SMBCLK pin is *Not* overdriven low. The other SMBus logic controls the state of the pin.		1b	RW
01	SMBDATA_CUR_STS	This bit has a default value that is dependent on an external signal level. This pin returns the value on the SMBDATA pin. This allows software to read the current state of the pin. 0 = Low 1 = High		1b	RO
00	SMBCLK_CUR_STS	This bit has a default value that is dependent on an external signal level. This pin returns the value on the SMBCLK pin. This allows software to read the current state of the pin. 0 = Low 1 = High		1b	RO

a. Reset by CF9 RESET or RSMRST#

24.4 Host Controller

24.4.1 Overview

The SMB Host Controller is used to send commands to other SMB slave devices. Software sets up the host controller with an address, command, and for writes, data and optionally PEC; and then tells the controller to start. When the controller has finished transmitting data on writes, or receiving data on reads, it generates an SMI# or interrupt, if enabled.

The host controller supports eight command protocols of the SMB interface (see the *System Management Bus (SMBus) Specification, Version 2.0*): Quick Command, Send Byte, Receive Byte, Write Byte/Word, Read Byte/Word, Process Call, Block Read, Block Write and Block write-block read process call.

The SMB Host Controller requires that the various data and command fields be setup for the type of command to be sent. When software sets the START bit, the SMB Host Controller performs the requested transaction and interrupt the processor (or generate an SMI#) when its finished. Once a START command has been issued, the values of the "active registers" (Host Control, Host Command, Transmit Slave Address, Data0, Data1) should not be changed or read until the interrupt status bit (INTR) has been set (indicating the completion of the command). Any register values needed for computation purposes should be saved prior to issuing of a new command, as the SMB Host Controller will update all registers while completing the new command.

The CMI supports slave functionality, including the Host Notify protocol, on the SMLink pins when in TCO compatible mode. Therefore, in order to be fully compliant with the *SMBus Specification* (which requires the Host Notify cycle), the SMLink and SMBus signals must be tied together externally.

Using the SMB Host Controller to send commands to the SMB slave port is not supported.

24.4.2 Command Protocols

In all of the following commands, the Host Status Register (offset 00h) is used to determine the progress of the command. While the command is in operation, the HOST_BUSY bit is set. If the command completes successfully, the INTR bit is set in the Host Status Register. If the device does not respond with an acknowledge, and the transaction times out, the DEV_ERR bit is set. If software sets the KILL bit in the Host Control Register while the command is running, the transaction stops and the FAILED bit is set after the CMI forces a timeout. In addition, if the KILL bit is set during the CRC cycle, both the CRCE and DEV_ERR bits are also set. When the KILL bit is set, the CMI aborts current transaction by asserting SMBCLK low for greater than the timeout period, asserts a STOP condition and then releases SMBCLK and SMBDATA. However, setting the KILL bit does not affect SMLINK or TCO transactions or causes the CMI to force a timeout if it is not performing a transaction.

24.4.2.1 Quick Command

When programmed for a quick command, the Transmit Slave Address Register is sent. Table 24-31 shows the order. The PEC byte is never appended to the Quick Protocol. Software must force the PEC_EN bit to '0' when performing the Quick Command for possible future enhancements. Also, Quick Command with I2C_EN set produces undefined results. Software must force the I2C_EN bit to 0 when running this command.

Table 24-31. Quick Protocol

Bit	Description
1	Start Condition
2–8	Slave Address - 7 bits
9	Read / Write Direction
10	Acknowledge from slave
11	Stop

24.4.2.2 Send Byte / Receive Byte

For the send byte command, the Transmit Slave Address and Device Command Registers are sent.

The Receive Byte is similar to a Send Byte, the only difference is the direction of data transfer. When programmed for the receive byte command, the Transmit Slave Address Register is sent. The data received is stored in the DATA0 register. The order sent/ received without PEC is shown in Table 24-32. Send Byte/Receive Byte command with I2C_EN set produces undefined results. Software must force the I2C_EN bit to 0 when running this command.

Table 24-32. Send / Receive Byte Protocol without PEC

Send Byte Protocol		Receive Byte Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Read
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Data byte from slave
19	Acknowledge from slave	19	NOT Acknowledge
20	Stop	20	Stop

The order sent/received, with PEC, is shown in Table 24-33.

Table 24-33. PEC Send / Receive Order

Send Byte Protocol		Receive Byte Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Read
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Data byte from slave
19	Acknowledge from slave	19	Acknowledge
20 – 27	PEC	20 – 27	PEC from slave
28	Acknowledge from slave	28	Not Acknowledge
29	Stop	29	Stop

24.4.2.3 Write Byte / Word

The first byte of a write byte/word access is the command code. The next one or two bytes are the data to be written. When programmed for a write byte/word command, the Transmit Slave Address, device command, and Data0 Registers are sent. In addition, the Data1 Register is sent on a write word command. The order of bits without PEC is shown in Table 24-34. Issuing a write byte/word command with I2C_EN set produces undefined results. Software must force the I2C_EN bit to 0 when running this command.

The order of bits with PEC is shown in Table 24-35.

Table 24-34. Write Byte / Word Protocol Without PEC

Write Byte Protocol		Write Word Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20 – 27	Data Byte - 8 bits	20 – 27	Data Byte Low - 8 bits
28	Acknowledge from Slave	28	Acknowledge from Slave
29	Stop	29 – 36	Data Byte High - 8 bits
		37	Acknowledge from slave
		38	Stop

Table 24-35. PEC Bit Order

Write Byte Protocol		Write Word Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20 – 27	Data Byte - 8 bits	20 – 27	Data Byte Low - 8 bits
28	Acknowledge from Slave	28	Acknowledge from Slave
29 – 36	PEC	29 – 36	Data Byte High - 8 bits
37	Acknowledge from Slave	37	Acknowledge from slave
38	Stop	38 – 45	PEC
		46	Acknowledge from slave
		47	Stop

24.4.2.4 Read Byte / Word

Reading data is slightly more complicated than writing data. First a command to the slave device must be written. Then it must follow that command with a repeated start condition to denote a read from that device's address. The slave then returns one or two bytes of data.

When programmed for the read byte/word command, the transmit slave address and device command registers are sent. Data is received into the DATA0 on the read byte, and the DAT0 and DATA1 registers on the read word. The order sent and received with PEC disabled is shown in Table 24-36.

Read byte/word command with I2C_EN set produces undefined results. Software must force the I2C_EN bit to 0 when running this command.

Table 24-36. Read Byte / Word Protocol without PEC

Read Byte Protocol		Read Word Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20	Repeated Start	20	Repeated Start
21 – 27	Slave Address - 7 bits	21 – 27	Slave Address - 7 bits
28	Read	28	Read
29	Acknowledge from slave	29	Acknowledge from slave
30 – 37	Data from slave - 8 bits	30 – 37	Data Byte Low from slave - 8 bits
38	NOT acknowledge	38	Acknowledge
39	Stop	39 – 46	Data Byte High from slave - 8 bits
		47	NOT acknowledge
		48	Stop

The order sent and received with PEC enabled is shown in Table 24-37.

Table 24-37. Read Byte / Word Protocol with PEC

Read Byte Protocol		Read Word Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20	Repeated Start	20	Repeated Start
21 – 27	Slave Address - 7 bits	21 – 27	Slave Address - 7 bits

Table 24-37. Read Byte / Word Protocol with PEC

28	Read	28	Read
29	Acknowledge from slave	29	Acknowledge from slave
30 – 37	Data from slave - 8 bits	30 – 37	Data Byte Low from slave - 8 bits
38	Acknowledge	38	Acknowledge
39 – 46	PEC from slave	39 – 46	Data Byte High from slave - 8 bits
47	NOT acknowledge	47	Acknowledge
48	Stop	48 – 55	PEC from slave
		56	NOT acknowledge
		57	Stop

24.4.2.5 Process Call

The process call is named because a command sends data and waits for the slave to return a value dependent on that data. The protocol is simply a write word followed by a read word, but without a second command or stop condition.

When programmed for the process call command, the CMI transmits the transmit address, device command, and DATA0 and DATA1 registers. Data received from the device is stored in the DATA0 and DATA1 registers. The value written into bit 0 of the Transmit Slave Address Register (SMBus Offset 04h) needs to programmed to 0.

Note: If the I2C_EN bit is set, then the command field is not sent.

The order sent with PEC disabled is shown in Table 24-38. The process call command with I2C_EN set and either the PEC_EN or AAC bit set produces undefined results. Software must either force the I2C_EN bit or both PEC_EN and AAC bits to 0 when running this command.

Table 24-38. Process Call Protocol without PEC

Bit	Description
1	Start
2 – 8	Slave Address - 7 bits
9	Write
10	Acknowledge from Slave
11 – 18	Command code - 8 bits (Skip if I2C_EN is set)
19	Acknowledge from slave (Skip if I2C_EN is set)
20 – 27	Data byte Low - 8 bits
28	Acknowledge from Slave
29 – 36	Data Byte High - 8 bits
37	Acknowledge from slave
38	Repeated Start
39 – 45	Slave Address - 7 bits
46	Read
47	Acknowledge from slave
48 – 55	Data Byte Low from slave - 8 bits
56	Acknowledge

Table 24-38. Process Call Protocol without PEC

57 – 64	Data Byte High from slave - 8 bits
65	NOT acknowledge
66	Stop

The order sent with PEC enabled is shown in Table 24-39.

Table 24-39. Process Call Protocol with PEC

Bit	Description
1	Start
2 – 8	Slave Address - 7 bits
9	Write
10	Acknowledge from Slave
11 – 18	Command code - 8 bits
19	Acknowledge from slave
20 – 27	Data byte Low - 8 bits
28	Acknowledge from Slave
29 – 36	Data Byte High - 8 bits
37	Acknowledge from slave
38	Repeated Start
39 – 45	Slave Address - 7 bits
46	Read
47	Acknowledge from slave
48 – 55	Data Byte Low from slave - 8 bits
56	Acknowledge
57 – 64	Data Byte High from slave - 8 bits
65	Acknowledge
66 – 73	PEC from slave
74	NOT acknowledge
75	Stop

24.4.2.6 Block Read/Write

The CMI contains a 32-byte buffer for read and write data which can be enabled by setting bit '1' of the Auxiliary Control register at offset 0Dh in I/O space, as opposed to a single byte of buffering. This 32-byte buffer is filled with write data before transmission and filled with read data on reception. In the CMI, the interrupt is generated only after a transmission or reception of 32 bytes, or when the entire byte count has been transmitted/received.

The block write command with I2C_EN set and either the PEC_EN or AAC bit set produces undefined results. Software must either force the I2C_EN bit or both PEC_EN and AAC bits to 0 when running this command.

24.4.2.6.1 SM Bus Mode

The block write begins with a slave address and a write condition. After the command code the CMI issues a byte count describing how many more bytes will follow in the message. If a slave had 20 bytes to send, the first byte would be the number 20 (14h), followed by 20 bytes of data. The byte count may not be 0. A block read or write is allowed to transfer a maximum of 32 data bytes.

When programmed for a block write command, the transmit slave address, device command, and Data0 (count) registers are sent. Data is then sent from the Block Data Byte register; the total data sent being the value stored in the Data0 Register. On block read commands, the first byte received is stored in the Data0 register, and the remaining bytes are stored in the Block Data Byte register.

24.4.2.6.2 I²C Mode

The format of the command changes slightly for block commands if the I2C_EN bit is set. The CMI still sends the number of bytes (on writes) or receive the number of bytes (on reads) indicated in the DATA0 register. However, it does not send the contents of the DATA0 register as part of the message.

The format of the command changes slightly for a block write if the I2C_EN bit is set. The CMI still sends the number of bytes indicated in the DATA0 register. However, it does not send the contents of the DATA0 register as part of the message.

The protocol for the block write and block read without PEC is shown in Table 24-40.

Table 24-40. Block Read/Write Protocol without PEC (Sheet 1 of 2)

Block Write Protocol		Block Read Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20 – 27	Byte Count - 8 bits (skip this step if I2C_EN bit set)	20	Repeated Start
28	Acknowledge from Slave (skip this step if I2C_EN bit set)	21 – 27	Slave Address - 7 bits
29 – 36	Data Byte 1 - 8 bits	28	Read
37	Acknowledge from Slave	29	Acknowledge from slave
38 – 45	Data Byte 2 - 8 bits	30 – 37	Byte Count from slave - 8 bits
46	Acknowledge from slave	38	Acknowledge
...	Data Bytes / Slave Acknowledges...	39 – 46	Data Byte 1 from slave - 8 bits
...	Data Byte N - 8 bits	47	Acknowledge
...	Acknowledge from Slave	48 – 55	Data Byte 2 from slave - 8 bits
...	Stop	56	Acknowledge
		...	Data Bytes from slave/Acknowledge

Table 24-40. Block Read/Write Protocol without PEC (Sheet 2 of 2)

		...	Data Byte N from slave - 8 bits
		...	NOT Acknowledge
		...	Stop

The protocol for the block write and block read with PEC is shown in Table 24-41.

The block write command with I2C_EN set and the PEC_EN bit set produces undefined results. Software must force the PEC_EN bit to 0 when running this command.

Table 24-41. Block Read/Write Protocol with PEC

Block Write Protocol		Block Read Protocol	
Bit	**Description**	**Bit**	**Description**
1	Start	1	Start
2 – 8	Slave Address - 7 bits	2 – 8	Slave Address - 7 bits
9	Write	9	Write
10	Acknowledge from slave	10	Acknowledge from slave
11 – 18	Command code - 8 bits	11 – 18	Command code - 8 bits
19	Acknowledge from slave	19	Acknowledge from slave
20 – 27	Byte Count - 8 bits	20	Repeated Start
28	Acknowledge from Slave	21 – 27	Slave Address - 7 bits
29 – 36	Data Byte 1 - 8 bits	28	Read
37	Acknowledge from Slave	29	Acknowledge from slave
38 – 45	Data Byte 2 - 8 bits	30 – 37	Byte Count from slave - 8 bits
46	Acknowledge from slave	38	Acknowledge
...	Data Bytes / Slave Acknowledges...	39 – 46	Data Byte 1 from slave - 8 bits
...	Data Byte N - 8 bits	47	Acknowledge
...	Acknowledge from Slave	48 – 55	Data Byte 2 from slave - 8 bits
...	PEC – 8 bits	56	Acknowledge
...	Acknowledge from Slave	...	Data Bytes from slave/Acknowledge
...	Stop	...	Data Byte N from slave - 8 bits
		...	Acknowledge
		...	PEC from slave – 8 bits
		...	NOT Acknowledge
		...	Stop

24.4.2.7 I^2C Read

This command allows the CMI to perform block reads to certain I2C devices, such as serial E2PROMs. The SMBus Block Read supports the 7-bit addressing mode only. However this doesn't allow access to devices that need to use the I^2C "Combined Format" that has data bytes after the address. Typically these data bytes correspond to an offset (address) within the serial memory chips.

The I^2C Read command with either PEC_EN or AAC bit set produces undefined results. Software must force both PEC_EN and AAC bits to 0 when running this command.

To support these devices, the CMI implements an I2C Read command with the following format:

Bit	Description
1	Start
2 – 8	Slave Address - 7 bits
9	Write
10	Acknowledge from slave
11 – 18	Send DATA1 register
19	Acknowledge from slave
20	Repeated Start
21 – 27	Slave Address - 7 bits
28	Read
29	Acknowledge from slave
30 – 37	Data Byte 1 from slave - 8 bits
38	Acknowledge
39 – 46	Data Byte 2 from slave - 8 bits
47	Acknowledge
…	Data Bytes from slave/Acknowledge
…	Data Byte N from slave - 8 bits
…	NOT Acknowledge
…	Stop

The CMI continues reading data from the peripheral until the NAK is received.

Note: This new command is supported independent of the setting of the I2C_EN bit.

Note: The value written into bit 0 of the Transmit Slave Address register (SMBus Offset 04h) must be 0.

24.4.2.8 Block Write-Block Read Process Call

The block write-block read process call is a two-part message. The call begins with a slave address and a write condition. After the command code the host issues a write byte count (M) that describes how many more bytes will be written in the first part of the message. If a master has six bytes to send, the byte count field will have the value 6 (0000 0110b), followed by the six bytes of data. The write byte count (M) cannot be zero.

The second part of the message is a block of read data beginning with a repeated start condition followed by the slave address and a read bit. The next byte is the read byte count (N), which may differ from the write byte count (M). The read byte count (N) cannot be zero.

The combined data payload must not exceed 32 bytes. The byte length restrictions of this process call are summarized as follows:

- M ≥ 1 byte
- N ≥ 1 byte
- M + N ≤ 32 bytes

The read byte count does not include the PEC byte. The PEC is computed on the total message beginning with the first slave address and using the normal PEC computational rules. It is highly recommended that a PEC byte be used with the block write-block read process call. Software must do a read to the command register (offset 2h) to reset the 32 byte buffer pointer prior to reading the block data register.

Note: There is no STOP condition before the repeated START condition, and that a NACK signifies the end of the read transfer.

Note: E32B in the Auxiliary Control register must be set when using this protocol.

Table 24-42. Block Write-Block Read Process Call Protocol with / without PEC

Bit	Description
1	Start
2 – 8	Slave Address - 7 bits
9	Write
10	Acknowledge from Slave
11 – 18	Command code - 8 bits
19	Acknowledge from slave
20 – 27	Data Byte Count (M) - 8 bits
28	Acknowledge from Slave
29 – 36	Data Byte (1) - 8 bits
37	Acknowledge from slave
38 – 45	Data Byte (2) - 8 bits
46	Acknowledge from slave
...	...
	Data Byte (M) - 8 bits
	Acknowledge from slave
	Repeated Start
	Slave Address - 7 bits
	Read
	Acknowledge from slave
	Data Byte Count (N) from slave – 8 bits
	Acknowledge from master
	Data Byte (1) from slave – 8 bits
	Acknowledge from master
	Data Byte (2) from slave – 8 bits
	Acknowledge from master
...	...
	Data Byte Count (N) from slave – 8 bits
	Acknowledge from master (Skip if no PEC)
	PEC from slave (Skip if no PEC)
	NOT acknowledge
	Stop

24.4.3 I²C Behavior

When the I2C_EN bit is set, the CMI SMBus logic is set to communicate with I²C devices. This forces the following changes:

1. The Process Call command skips the command code (and its associated acknowledge)
2. The Block Write command skips sending the byte count (DATA0)

In addition, the CMI supports the I²C Read command. This is independent of the I2C_EN bit. When operating in I²C mode, (I2C_EN bit set), the CMI never uses the 32-byte buffer for any block commands.

24.4.4 Heartbeat for Use with External LAN

This method allows the CMI to send messages to an External LAN controller when the processor is otherwise unable to do so. It uses the SMLink Interface between the CMI and external LAN controller in TCO compatible mode. The actual heartbeat message is a block write. Only eight bytes are sent.

See Chapter 18.0, “System Management,” for more details on the heartbeat packet format, and the specific bits sent in the packet.

24.5 Bus Arbitration

Several masters may attempt to get on the bus at the same time by driving the SMBDATA line low to signal a start condition. The CMI continuously monitors the SMBDATA line. When the CMI is attempting to drive the bus to a ‘1’ by letting go of the SMBDATA line, and it samples SMBDATA low, then some other master is driving the bus and the CMI stops transferring data.

If the CMI detects loss of arbitration, the condition is called a collision. The CMI sets the BUS_ERR bit in the Host Status register, and if enabled, generates an interrupt or SMI#. The processor is responsible for restarting the transaction.

24.6 Bus Timings

The SM Bus runs at between 10 – 100 kHz. Most of the timings associated with the SM Bus are microseconds in length. The SM Bus runs off of a divide by two of the RTC clock internally and employs counters of various length off of the RTC clock to drive the SM Bus.

When the CMI is a SM Bus master, it drives the clock. When the CMI is sending address or command as an SM Bus master, or data bytes as a master on writes, it will drive data relative to the clock it is also driving. It does not start toggling the clock until the start or stop condition meets proper setup and hold. The CMI also guarantees minimum time between SM Bus transactions as a master.

24.6.1 Clock Stretching

Some devices may not be able to handle their clock toggling at the rate that the CMI as an SM Bus master would like. They have the capability of stretching the low time of the clock. When the CMI attempts to release the clock (allowing the clock to go high), the clock remains low for an extended period of time.

The CMI monitors the SM Bus clock line after it releases the bus to determine whether to enable the counter for the high time of the clock. While the bus is still low, the high time counter must not be enabled. Similarly, the low period of the clock can be stretched by an SM Bus master if it is not ready to send or receive data.

24.6.2 Bus Time Out (CMI as SMB Master)

If there is an error in the transaction, such that an SMBus device does not signal an acknowledge, or holds the clock lower than the allowed time-out time, the transaction will time out. The CMI discards the cycle, and set the DEV_ERR bit. The time out minimum is 25 ms. The time-out counter inside the CMI starts when the first bit of data is transferred by the CMI. The time-out minimum is 25 ms (800 RTC clocks).

The 25 ms timeout counter does not count under the following conditions:

1. BYTE_DONE_STATUS bit (SMBus I/O Offset 00h, bit 7) is set, and
2. The SECOND_TO_STS bit (TCO I/O Offset 06h, bit 1) is not set (this indicates that the system has not locked up).

24.7 Interrupts/SMI#

The CMI SM Bus controller uses PIRQB# as its interrupt pin. However, the system can alternatively be set up to generate SMI# instead of an interrupt, by setting the SMBUS_SMI_EN bit (Device 31, Function 0, Offset 40h, bit 1).

The following tables specify how the various enable bits in the SMBus function control the generation of the interrupt, Host and Slave SMI, and Wake internal signals. The rows in the tables are additive, which means that if more than one row is true for a particular scenario then the results for all of the activated rows occurs.

Table 24-43. Summary of Enables for SMBALERT#

Event	INTREN (Host Control I/O Register, Offset 02h, Bit 0)	SMB_SMI_EN (Host Configuration Register, D31, F3, Offset 40h, Bit 1)	SMBALERT_DIS (Slave Command I/O Register, Offset 11h, Bit 2)	Result
SMBALERT# asserted low (always reported in SMBALERT_STS- Host Status Register, bit 5)	X	X	X	Wake generated
	X	1	0	Slave SMI# generated (SMBUS_SMI_STS)
	1	0	0	Interrupt generated

Table 24-44. Summary of Enables for SMBus Slave Write, and SMBus Host Events

Event	INTREN (Host Control I/O Register, Offset 02h, Bit 0)	SMB_SMI_EN (Host Configuration Register, D31, F3, Offset 40h, Bit 1)	Result
Slave Write to Wake/SMI# command	X	X	Wake generated when asleep Slave SMI# generated when awake (SMBUS_SMI_STS)
Slave Write to SMLINK_SLAVE_SMI command	X	X	Slave SMI# generated when in the S0 state (SMBUS_SMI_STS)
Any combination of Host Status Register [04:01] asserted	0	X	None
	1	0	Interrupt generated
	1	1	Host SMI# generated

Table 24-45. Summary of Enables for the Host Notify Command

HOST_NOTIFY_INTREN (Slave Control I/O Register, Offset 11h, bit 0)	SMB_SMI_EN (Host Configuration Register, D31, F3, Off40h, Bit 1)	HOST_NOTIFY_WKEN (Slave Control I/O Register, Offset 11h, bit 1)	Result
0	X	0	None
X	X	1	Wake generated
1	0	X	Interrupt generated
1	1	X	Slave SMI# generated (SMBUS_SMI_STS)

24.8 CRC Generation and Checking

If the AAC bit is set in the Auxiliary Control register, the CMI automatically calculates and drives CRC at the end of the transmitted packet for write cycles, and checks the CRC for read cycles. It does not transmit the contents of the PEC register for CRC. The PEC bit must not be set in the Host Control register if this bit is set, or unspecified behavior results.

If the read cycle results in a CRC error, the DEV_ERR bit and the CRCE bit in the Auxiliary Status register at offset 0Ch is set.

24.8.1 Slave Interface I/O Space

The following registers are used by the SMBus Slave logic. Refer to Section 24.3 for the complete list of SMB I/O Registers.

Table 24-46. Bus 0, Device 31, Function 3, Slave PCI Registers Mapped Through SM_Base (IO)

Offset Start	Offset End	Register ID - Description	Default Value
09h	09h	"Offset 09h: RSA: Receive Slave Address Register" on page 931	44h
0Ah	0Bh	"Offset 0Ah: SD: Slave Data Register" on page 931	0000h
10h	10h	"Offset 10h: SSTS: Slave Status Register" on page 932	00h
11h	11h	"Offset 11h: SCMD: Slave Command Register" on page 932	00h
14h	14h	"Offset 14h: NDA: Notify Device Address Register" on page 933	00h
16h	16h	"Offset 16h: NDLB: Notify Data Low Byte Register" on page 934	00h
17h	17h	"Offset 17h: NDHB: Notify Data High Byte Register" on page 934	00h

24.8.2 Register Details

24.8.2.1 Offset 09h: RSA: Receive Slave Address Register

Table 24-47. Offset 09h: RSA: Receive Slave Address Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 09h **Offset End:** 09h	
Size: 8 bit	**Default:** 44h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0b	
06 : 00	RSA	**SLAVE_ADDR[06:00]:** This field is the slave address that is decoded for read and write cycles. The default is not 0 so that it can respond even before the processor comes up (or if the processor is dead). This register is reset by CF9 RESET or RSMRST#, but not by PLTRST#.		1000100b	RW

24.8.2.2 Offset 0Ah: SD: Slave Data Register

Table 24-48. Offset 0Ah: SD: Slave Data Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 0Ah **Offset End:** 0Bh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SD	**SLAVE_DATA[15:00]:** This field is the 16-bit data value written by the external SMBus master. The processor can then read the value from this register. This register is reset by CF9 RESET or RSMRST#, but not by PLTRST#. SLAVE_DATA[07:00] corresponds to the Data Message Byte 0 (see Section 24.8.2.1) at Slave Write Register 4 in the table. SLAVE_[15:08] corresponds to the Data Message Byte 1 (see Section 24.8.2.1) at Slave Write Register 5 in the table.		0h	RO

24.8.2.3 Offset 10h: SSTS: Slave Status Register

Table 24-49. Offset 10h: SSTS: Slave Status Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 10h **Offset End:** 10h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved	Reserved		0h	
00	HOST_ NOTIFY_ STS	Software reads this bit to determine that the source of the interrupt or SMI# was the reception of the Host Notify Command. Software clears this bit after reading any information needed from the notify address and data registers by writing a 1 to this bit. The CMI allows the notify address and data registers to be overwritten once this bit has been cleared. When this bit is 1,the CMI will NACK the first byte (host address) of any new “Host Notify” commands on the SMLink. Writing a 0 to this bit has no effect 0 = Bit is clear, allows the notify address and data registers to be overwritten. 1 = This bit is set when the CMI has completely received a successful Host Notify Command on the SMLink pins.		0h	RWC

a. This register is in the resume well and is reset by CF9 RESET or RSMRST#. All bits in this register are implemented in the 64 kHz clock domain. Therefore, software must poll the register until a write takes effect before assuming that a write has completed internally.

24.8.2.4 Offset 11h: SCMD: Slave Command Register

Table 24-50. Offset 11h: SCMD: Slave Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 11h **Offset End:** 11h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		00h	
02	SMBALERT_DIS	0 = Allows the generation of interrupt or SMI#. 1 = Software sets this bit to 1 to block the generation of the interrupt or SMI# due to the SMBALERT# source. This bit is logically inverted and ANDed with the SMBALERT_STS bit. The resulting signal is distributed to the SMI# and/or interrupt generation logic. This bit does not effect the wake logic.		0h	RW
01	HOST_ NOTIFY_ WKEN	Software sets this bit to 1 to enable the reception of a Host Notify command as a wake event. When enabled this event is ORed with the other SMBus wake events and is reflected in the SMB_WAK_STS bit of the General Purpose Event 0 Status register. 0 = Disable 1 = Enable		0h	RW

Table 24-50. Offset 11h: SCMD: Slave Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** SM_BASE (IO)		**Bus:Device:Function:** 0:31:3	**Offset Start:** 11h **Offset End:** 11h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
00	HOST_ NOTIFY_ INTREN	Software sets this bit to 1 to enable the generation of interrupt or SMI# when HOST_NOTIFY_STS is 1. This enable does not affect the setting of the HOST_NOTIFY_STS bit. When the interrupt is generated, either PIRQB or SMI# is generated, depending on the value of the SMB_SMI_EN bit (D31, F3, Off40h, B1). If the HOST_NOTIFY_STS bit is set when this bit is written to a 1, then the interrupt (or SMI#) will be generated. The interrupt (or SMI#) is logically generated by ANDing the STS and INTREN bits. 0 = Disable 1 = Enable		0h	RW

a. This register is in the resume well and is reset by CF9 RESET or RSMRST#. All bits in this register are implemented in the 64 kHz clock domain. Therefore, software must poll the register until a write takes effect before assuming that a write has completed internally.

24.8.2.5 Offset 14h: NDA: Notify Device Address Register

This register is in the resume well and is reset by CF9 RESET or RSMRST#.

Table 24-51. Offset 14h: NDA: Notify Device Address Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)		**Bus:Device:Function:** 0:31:3	**Offset Start:** 14h **Offset End:** 14h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	DEVICE_ ADDRESS	This field contains the 7-bit device address received during the Host Notify protocol of the *SMBus Specification*. Software should only consider this field valid when the HOST_NOTIFY_STS bit is set to 1.		0000000h	RO
00	Reserved	Reserved		0h	

a. Reset by CF9 RESET or RSMRST#.

24.8.2.6 Offset 16h: NDLB: Notify Data Low Byte Register

This register is in the resume well and is reset by CF9 RESET or RSMRST#.

Table 24-52. Offset 16h: NDLB: Notify Data Low Byte Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 16h **Offset End:** 16h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DATA_LOW_BYTE	This field contains the first (low) byte of data received during the Host Notify protocol of the *SMBus Specification*. Software should only consider this field valid when the HOST_NOTIFY_STS bit is set to 1.		00h	RO

a. Reset by CF9 RESET or RSMRST#.

24.8.2.7 Offset 17h: NDHB: Notify Data High Byte Register

This register is in the resume well and is reset by CF9 RESET or RSMRST#.

Table 24-53. Offset 17h: NDHB: Notify Data High Byte Register

Description:					
View: PCI	**BAR:** SM_BASE (IO)	**Bus:Device:Function:** 0:31:3		**Offset Start:** 17h **Offset End:** 17h	
Size: 8 bit	**Default:** 00h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	DATA_HIGH_BYTE	This field contains the second (high) byte of data received during the Host Notify protocol of the *SMBus Specification*. Software should only consider this field valid when the HOST_NOTIFY_STS bit is set to 1.		00h	RO

a. Reset by CF9 RESET or RSMRST#.

24.9 Slave Interface Behavioral Description

The SMBus slave logic does not generate or handle receiving the PEC byte. There is no ASF support.

The slave interface allows the CMI to decode cycles on SMLink in TCO compatible mode, and allows an external microcontroller to perform specific actions. Key features and capabilities:

- Supports decode of three types of messages: byte write, byte read, and host notify
- Register for the receive slave address. This is the address that the CMI decodes. A default value is provided so that the slave interface can be used without the processor having to program this register.
- Receive Slave Data register in the SMBus I/O space that includes the data written by the external microcontroller
- Registers that the external microcontroller can read to get the state
- Status bits to indicate that the SMLink/SMBus slave logic caused an interrupt or SMI#
  - — Bit 0 of the slave status register for the host notify command
  - — Bit 16 of the SMI Status Register for all others

Note: The external microcontroller should not attempt to access the SMBus slave logic until one second after both: RTEST# is high and RSMRST# is high.

If a master leaves the clock and data bits of the SMLink or SMBus interface at '1' for 50 µs or more in the middle of a cycle, the slave logic's behavior is undefined. This is interpreted as an unexpected idle and should be avoided when performing management activities to the slave logic.

24.9.1 Format of Slave Write Cycle

The external master performs byte write commands to the SMBus Slave Interface. The Command field (bits 11 – 18) indicate which register is being accessed. The Data field (bits 20 – 27) indicate the value that should be written to that register.

The write cycle format is shown below in Table 24-54. Table 24-55 has the values associated with the registers.

Table 24-54. Slave Write Cycle Format

Bit	Description	Driven By	Comment
1	Start Condition	External Microcontroller	
2 — 8	Slave Address - 7 bits	External Microcontroller	Must match value in Receive Slave Address register
9	Write	External Microcontroller	Hardwired to 0
10	ACK	CMI	
11 — 18	Command	External Microcontroller	This field indicates which register will be accessed. See Table 24-55 for the register definitions
19	ACK	CMI	
20 — 27	Register Data	External Microcontroller	See Table 24-55 for the register definitions
28	ACK	CMI	
29	Stop	External Microcontroller	

Table 24-55. Slave Write Registers

Register	Function:
0	Command Register. See Table 24-56 for legal values written to this register.
1 — 3	Reserved
4	Data Message Byte 0
5	Data Message Byte 1
6 — 7	Reserved
8	Reserved
9 — FFh	Reserved

Note: The external microcontroller is responsible to make sure that it does not update the contents of the data byte registers until they have been read by the system processor. The CMI overwrites the old value with any new value received. A race condition is possible where the new value is being written to the register just at the time it is being read. The CMI does not attempt to cover this race condition (i.e., unpredictable results in this case).

Table 24-56. Command Types

Command Type	Description
0	Reserved
1	**WAKE/SMI#:** Wake system if it is not already awake. If system is already awake, then an SMI# is generated.
2	**Unconditional Powerdown:** This command should set the PWRBTNOR_STS bit, and have the same effect as the power button override occurring.
3	**HARD RESET Without Power Cycling SYSTEM:** The causes a soft reset of the system (does not include cycling of the power supply). This is equivalent to a write to the CF9h register with bits 02:01 set to 1, but bit 03 set to 0.
4	**HARD RESET SYSTEM:** The causes a hard reset of the system (including cycling of the power supply). This is equivalent to a write to the CF9h register with bits 03:01 set to 1.
5	**Disable the TCO Messages.** This command disables the IICH from sending Heartbeat and Event messages. Once this command has been done, there is no method to reenable the Heartbeat and Event messages, until CF9 RESET or RSMRST# goes low and then high.
6	**WD RELOAD:** Reload watchdog timer.
7	Reserved
8	**SMLINK_SLAVE_SMI:** When the CMI detects this command type while in the S0 state, it will set the SMLINK_SLAVE_SMI_STS bit. This command should only be used if the system is in an S0 state. If the message is received during S3 or S5 states, it is acknowledged by the CMI but the SMLINK_SLAVE_SMI_STS bit is not set. ***Note:*** It is possible that the system transitions out of the S0 state at the same time that the SMLINK_SLAVE_SMI command is received. In this case, the SMLINK_SLAVE_SMI_STS bit may get set but not serviced before the system goes to sleep. Once the system returns to S0, the SMI associated with this bit would then be generated. Software must be able to handle this scenario.
9 — FFh	Reserved

24.9.2 Format of Read Command

The external master performs byte read commands to the SMBus Slave Interface. The Command field (bits 11-18) indicate which register is being accessed. The Data field (bits 30-37) indicate the value that should be read from that register. Table 24-57 shows the read cycle format. Table 24-58 shows the register mapping for the data byte.

Table 24-57. Slave Read Cycle Format

Bit	Description	Driven by:	Comment:
1	Start	External Microcontroller	
2 — 8	Slave Address - 7 bits	External Microcontroller	Must match value in Receive Slave Address register
9	Write	External Microcontroller	Hardwired to 0
10	ACK	CMI	
11 — 18	Command code - 8 bits	External Microcontroller	Indicates which register is being accessed. See Table 24-58 for list of implemented registers.
19	ACK	CMI	
20	Repeated Start	External Microcontroller	
21 — 27	Slave Address - 7 bits	External Microcontroller	Must match value in Receive Slave Address register
28	Read	External Microcontroller	Hardwired to 1
29	ACK	CMI	
30 — 37	Data Byte	CMI	Value depends on register being accessed. See Table 24-58 for list of implemented registers.
38	NOT ACK	External Microcontroller	
39	Stop	External Microcontroller	

Table 24-58. Data Values for Slave Read Registers (Sheet 1 of 2)

Register	Bits	Description
0	07:00	Reserved for capabilities indication. Should always return 00h. Future chips may return another value to indicate different capabilities.
1	02:00	System Power State 000 = S0 001 = Reserved 010 = Reserved 011 = S3 100 = Reserved 101 = S5 110 = Reserved 111 = Reserved
	07:03	Reserved
2	03:00	Reserved
	07:04	Reserved
3	05:00	Watchdog Timer current value. Watchdog Timer has 10 bits, but this field is only 6 bits. If the current value is greater than 3Fh, the CMI always reports 3Fh in this field.
	07:06	Reserved

Table 24-58. Data Values for Slave Read Registers (Sheet 2 of 2)

Register	Bits	Description
4	00	1 = The Intruder Detect (INTRD_DET) bit is set. This indicates that the system cover has been opened.
	01	1 = BTI Temperature Event occurred. This bit is set if the CMI's PROCHOT# input signal is active. Need to take after polarity control.
	02	DOA processor Status. This bit is 1 to indicate that the processor is dead.
	03	1 = SECOND_TO_STS bit set. This bit is set after the second timeout (SECOND_TO_STS bit) of the Watchdog Timer occurs.
	06:04	Reserved. Will always be 0, but software should ignore.
	07	Reflects the value of the GPI[11]/SMBALERT# pin (and is dependent upon the value of the GPI_INV[11] bit. If the GPI_INV[11] bit is 1, then the value in this bit equals the level of the GPI[11]/SMBALERT# pin (high = 1, low = 0). If the GPI_INV[11] bit is 0, then the value of this bit equals the inverse of the level of the GPI[11]/SMBALERT# pin (high = 0, low = 1).
5	00	FWH bad bit. This bit is 1 to indicate that the FWH read returned FFh, which indicates that it is probably blank.
	01	Battery Low Status. '1' if the BATLOW# pin is a '0'.
	02	CPU Power Failure Status: '1' if the CPUPWR_FLR bit in the GEN_PMCON_2 register is set.
	07:01	Reserved
6	07:00	Contents of the Message 1 register. See Section 18.2.2.8 for details.
7	07:00	Contents of the Message 2 register. See Section 18.2.2.8 for details.
8	07:00	Contents of the WDSTATUS register. See Section 18.2.2.9 for details.
9	07:00	Contents of the SATA SGPIO Control Register (bits 07:00). See Section 48.3.3.1 for details.
A	07:00	Contents of the SATA SGPIO Control Register (bits 15:08). See Section 48.3.3.1 for details.
B	07:00	Contents of the SATA SGPIO Control Register (bits 23:16). See Section 48.3.3.1 for details.
C – FFh	07:00	Reserved

Warning: The external microcontroller is responsible to make sure that it does not read the contents of the various message registers until they have been written by the system processor. The CMI overwrites the old value with any new value received. A race condition is possible where the new value is being written to the register just at the time it is being read. The CMI does not attempt to cover this race condition (i.e., unpredictable results in this case).

Behavioral Notes:

The SMBus protocol always has either start bit-address-write bit or repeated start bit-address-read bit. The CMI is implemented such that the read/write bit in the repeated start phase is ignored with an assumption that the protocol always followed. In other words, if start-address-read occurs (which is illegal for SMBus byte read protocol), the CMI still grabs the cycle. In another case, if a repeated start-address-write sequence occurs, then the cycle continues as a slave read.

24.9.3 Format of the Host Notify Command

The CMI tracks and responds to the standard Host Notify command as specified in the *SMBus Specification*. The host address for this command is fixed to 0001000b. If the CMI already has data for a previously-received host notify command which has not been serviced yet by the host software (as indicated by the HOST_NOTIFY_STS bit), then it will NACK following the host address byte of the protocol. This allows the host to communicate non-acceptance to the master and retain the host notify address and data values for the previous cycle until host software completely services the interrupt.

Note: Host software must always clear the HOST_NOTIFY_STS bit *after* completing any necessary reads of the address and data registers.

Table 24-59 shows the host notify protocol.

Table 24-59. Host Notify Protocol

Bit	Description	Driven by:	Comment:
1	Start	External Master	
2 – 8	SMB Host Addr - 7 bits	External Master	Always 0001_000
9	Write	External Master	Hardwired to 0
10	ACK (or NACK)	CMI	The CMI NACKs if HOST_NOTIFY_STS is 1
11 – 17	Device Address - 7 bits	External Master	Indicates the address of the master; loaded in to the Notify Device Address Register
18	Unused- Hardwired to 0	External Master	7-bit-only address; this bit is inserted to complete the byte
19	ACK	CMI	
20 – 27	Data Byte Low	External Master	Loaded in to the Notify Data Low Byte Register
28	ACK	CMI	
29 – 36	Data Byte High	External Master	Loaded in to the Notify Data High Byte Register
37	ACK	CMI	
38	Stop	External Master	

§ §

25.0 USB (1.1) Controller: Bus 0, Device 29, Function 0

25.1 USB (1.1) Controller Configuration Register Details

Note: Reserved bits are read only.

Warning: Address locations that are not listed are considered reserved register locations and are read only. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure and unpredictable results.

Table 25-1. Bus 0, Device 29, Functions 0, Summary of USB (1.1) Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	"ID - Identifiers Register" on page 942	50338086h
04h	05h	"PCICMD - Command Register" on page 942	0000h
06h	07h	"PCISTS - Device Status Register" on page 943	0280h
08h	08h	"RID - Revision ID Register" on page 944	Variable
0Ah	0Ah	"SUBC - Sub Class Code Register" on page 945	03h
0Bh	0Bh	"BCC - Base Class Code Register" on page 945	0Ch
0Dh	0Dh	"MLT - Master Latency Timer Register" on page 945	00h
0Eh	0Eh	"HDR - Header Type Register" on page 946	Variable
20h	23h	"USBIOBAR - Base Address Register" on page 946	00000001h
2Ch	2Dh	"USBx_SVID - USB Subsystem Vendor ID Register" on page 947	0000h
2Eh	2Fh	"USBx_SID - USB Subsystem ID Register" on page 947	0000h
3Ch	3Ch	"INTL - Interrupt Line Register" on page 948	00h
3Dh	3Dh	"INTP - Interrupt Pin Register" on page 948	Variable
60h	60h	"SBRN - Serial Bus Release Number Register" on page 948	10h
C0h	C1h	"USBLKMCR - USB Legacy Keyboard/Mouse Control Register" on page 949	2000h
C4h	C4h	"USBREN - USB Resume Enable Register" on page 951	00h
C8h	C8h	"USBCWP - USB Core Well Policy Register" on page 951	00h
F8h	FBh	"MANID - Manufacturer ID Register" on page 952	00010F90h

25.1.1 Register Details

25.1.1.1 ID - Identifiers Register

Table 25-2. ID - Identifiers Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 00h **Offset End:** 03h	
Size: 32 bit	**Default:**	50338086h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	DID	**Device ID:** This 16-bit field is defined as follows: Function 0 5033h		5033h	RO
15 : 00	VID	**Vendor ID:** 16-bit field which indicates the company vendor as Intel.		8086h	RO

25.1.1.2 PCICMD - Command Register

Table 25-3. PCICMD - Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	
10	INTDIS	**Interrupt Disable:** 0 = Enable. The function is able to generate its interrupt to the interrupt controller. 1 = Disable. The function is not capable of generating interrupts. The corresponding Interrupt Status bit is not affected by the interrupt enable.		0h	RW
09	FBE	**Fast Back to Back Enable:** Hardwired to‘0’.		0h	
08	SERREN	**SERR# Enable:** Hardwired to‘0’.		0h	
07	WCC	**Wait Cycle Control:** Hardwired to‘0’.		0h	
06	PER	**Parity Error Response:** Hardwired to‘0’.		0h	
05	VGAPS	**VGA Palette Snoop:** Hardwired to‘0’.		0h	
04	PMWE	**Postable Memory Write Enable:** Hardwired to‘0’.		0h	
03	SCE	**Special Cycle Enable:** Hardwired to‘0’.		0h	

Table 25-3. PCICMD - Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	BME	**Bus Master Enable:** 0 = Disable. 1 = Enable. Can generate cycles to main memory as a master for USB transfers.		0h	RW
01	MSE	**Memory Space Enable:** Hardwired to‘0’.		0h	
00	IOSE	**I/O Space Enable:** This bit controls access to the I/O space registers. 0 = Disable: Accesses to the USB I/O registers is Disabled. 1 = Accesses to the USB I/O registers is enabled. The Base Address register for USB must be programmed before this bit is set.		0h	RW

25.1.1.3 PCISTS - Device Status Register

Table 25-4. PCISTS - Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0280h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** 0 = No parity error detected. 1 = Set when a data parity error is detected on writes to the UHCI register space or on read completions returned to the host controller. ***Note:*** Software sets this bit to 0 by writing a 1 to this bit location.		0h	RWC
14	SSE	Reserved		0h	
13	RMA	**Received Master-Abort Status:** 0 = No master abort generated by USB. 1 = USB as a master, receives a master abort. ***Note:*** Software sets this bit to 0 by writing a 1 to this bit location.		0h	RWC
12	Reserved	Reserved		0h	
11	STA	**Signaled Target-Abort Status:** 0 = Did not terminate a transaction to a USB function with a target abort. 1 = USB function is targeted with a transaction that terminates with a target abort. ***Note:*** Software sets this bit to 0 by writing a 1 to this bit location.		0h	RWC

Table 25-4. PCISTS - Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	0280h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
10 : 09	DEVT	**DEVSEL# Timing Status:** This 2-bit field defines the timing for DEVSEL# assertion. These read-only bits indicate the DEVSEL# timing when performing a positive decode. Since USB is not PCI-based, the value in this field is arbitrary.		01h	RO
08	Reserved	Hardwired as‘0’.		0h	
07	Reserved	Hardwired as‘1’		1	
06	Reserved	Hardwired as‘0’.		0h	
05	Reserved	Hardwired as‘0’.		0h	
04	Reserved	Hardwired as‘0’.		0h	
03	INTSTAT	**Interrupt Status:** This read-only bit reflects the state of this function’s interrupt at the input of the enable/disable logic. 0 = Interrupt is deasserted. 1 = Interrupt is asserted. The value reported in this bit is independent of the value in the Interrupt Enable bit.		0h	RO
02 : 00	Reserved	Reserved		000h	

25.1.1.4 RID - Revision ID Register

The value reported in this register depends on the value written to the Revision ID in Device 31, Function 0, Offset 08h.

Table 25-5. RID - Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:**	Variable		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	REVID	**Revision ID:** 8-bit value that indicates the revision number of the USB1.1 interface.The value reported in this register depends on the value written to the Revision ID in Device 31, Function 0, Offset 08h. This register follows the ICH revision ID scheme as defined in Section 19.2.1.4, “Offset 08h: RID - Revision ID Register” on page 736.		Variable	RO

25.1.1.5 SUBC - Sub Class Code Register

A value of 03h indicates that this is a Universal Serial Bus Host Controller.

Table 25-6. SUBC - Sub Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:**	03h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCC	**Sub-Class Code:** Indicates the device is a USB1.1 device.		03h	RO

25.1.1.6 BCC - Base Class Code Register

A value of 0Ch indicates that this is a Serial Bus controller.

Table 25-7. BCC - Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:**	0Ch		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	BCC	**Base Class Code:** Indicates the device is a USB device.		0Ch	RO

25.1.1.7 MLT - Master Latency Timer Register

Because the USB controller is internally implemented with arbitration on the CMI, it does not need a master latency timer. The bits are hardwired to 0.

Table 25-8. MLT - Master Latency Timer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:**	00h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MLC	**Master Latency Count:** Hardwired to 0.		0h	RO

25.1.1.8 HDR - Header Type Register

For function 0, bit 7 is determined by the values in the USB Function Disable bits (11:8 of the Function Disable register in Memory-mapped configuration space).

Table 25-9. HDR - Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:**	Variable		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	MFB	**Multi-Function Bit:** 0 = Single-function device. 1 = Multi-function device. Since the upper functions in this device can be individually hidden, this bit must be based on the function-disable bits in the Function Disable register in the memory-mapped configuration space as follows: D29_F7_Disable (bit 15) / D29_F0_Disable (bit 8) / Multi-Function Bit 0 / 0 / 1 0 / 1 / Invalid 1 / 0 / 0 1 / 1 / USB Disabled ***Note:***		Variable	RO
06 : 00	CONFIGL	**Configuration Layout:** Hardwired to 00h, which indicates the standard PCI configuration layout.		00h	RO

25.1.1.9 USBIOBAR - Base Address Register

USBIOBAR sets the base address in I/O space for additional USB configuration registers (see Section 25.2, “USB (1.1) Controller I/O-Mapped Register Details” on page 953). These registers can be mapped anywhere in the 64K I/O space on 32-byte boundaries.

Table 25-10. USBIOBAR - Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 20h **Offset End:** 23h	
Size: 32 bit	**Default:**	00000001h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Reserved	Reserved		0h	
15 : 05	BA	**Base Address:** Bits [15:5] correspond to I/O address signals AD [15:5], respectively. This gives 32 bytes of relocatable I/O space.		0h	RW
04 : 01	Reserved	Reserved		0h	
00	RTE	**Resource Type Indicator:** This bit is hardwired to 1 indicating that the base address field in this register maps to I/O space		1	RO

25.1.1.10 USBx_SVID - USB Subsystem Vendor ID Register

Note: x indicates the USB controller number.

BIOS sets the value in this register to identify the Subsystem Vendor ID. The USB_SVID register, in combination with the USB Subsystem ID register, enables the operating system to distinguish each subsystem from the others.

Note: Software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.

Table 25-11. USBx_SVID - USB Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** BIOS sets the value in this register to identify the Subsystem Vendor ID. The USB_SVID register, in combination with the USB Subsystem ID register, enables the operating system to distinguish each subsystem from the others.		0000h	RWO

25.1.1.11 USBx_SID - USB Subsystem ID Register

Note: x indicates the USB controller number.

BIOS sets the value in this register to identify the Subsystem ID. The SID register, in combination with the SVID register, enables the operating system to distinguish each subsystem from other(s).

Note: The software can write to this register only once per core well reset. Writes must be done as a single 16-bit cycle.

Table 25-12. USBx_SID - USB Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:**	0000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SUBID	**Subsystem ID:** BIOS sets the value in this register to identify the Subsystem ID. The SID register, in combination with the SVID register, enables the operating system to distinguish each subsystem from other(s). Indicates the subsystem as identified by the vendor. This field is write once and is locked until a reset occurs.		0000h	RWO

25.1.1.12 INTL - Interrupt Line Register

Table 25-13. INTL - Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTL	**Interrupt line:** This data is not used. This data is used to communicate to software which interrupt line the interrupt pin is connected to.		00h	RW

25.1.1.13 INTP - Interrupt Pin Register

Table 25-14. INTP - Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	INTPN	**Interrupt Pin:** This read-only value tells the software which interrupt pin each USB host controller uses. The upper 4 bits are hardwired to 0000b; the lower 4 bits are determine by the Interrupt Pin default values that are programmed in the memory-mapped configuration space as follows: Function 0 D29IP.U0P ***Note:*** This does not determine the mapping to the PIRQ pins.		Variable	RO

25.1.1.14 SBRN - Serial Bus Release Number Register

A value of 10h indicates that this is controller follows USB release 1.1.

Table 25-15. SBRN - Serial Bus Release Number Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** 60h **Offset End:** 60h	
Size: 8 bit	**Default:** 10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBN	Indicates that this controller follows USB release 1.0.		10h	RO

25.1.1.15 USBLKMCR - USB Legacy Keyboard / Mouse Control Register

This register is implemented separately in each of the USB1.1 functions. However, the enable and status bits for the trapping logic are ORed and shared, respectively, since their functionality is not specific to any one host controller.

Table 25-16. USBLKMCR - USB Legacy Keyboard / Mouse Control Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** C0h **Offset End:** C1h	
Size: 16 bit	**Default:**	2000h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	SMIBYENDPS	**SMI Caused by End of Pass-through:** Indicates if the event occurred. Even if the corresponding enable bit is not set in the Bit 7, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. 0 = No event occurred. 1 = Event occurred. ***Note:*** Writing a 1 to this bit (in any of the controllers) will clear the latch.		0h	RWC
14	Reserved	Reserved		0h	
13	USBPIRQEN	**PCI Interrupt Enable:** Used to prevent the USB controller from generating an interrupt due to transactions on its ports. When disabled, that it will probably be configured to generate an SMI using bit 4 of this register. Defaults to 1 for compatibility with older USB software. 0 = Disable 1 = Enable		1	RW
12	SMIBYUSB	**SMI Caused by USB Interrupt:** This bit indicates if an interrupt event occurred from this classic controller. The interrupts from the classic USB controller is taken before the enable in bit 13 has any effect to create this read-only bit. Even if the corresponding enable bit is not set in the Bit 4, then this bit may still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. 0 = No event occurred 1 = Event occurred. ***Note:*** Writing a 1 to this bit will have no effect. The software must clear the interrupts via the USB controllers.		0h	RO
11	TRAPBY64W	**SMI Caused by Port 64 Write:** Indicates if the event occurred. Even if the corresponding enable bit is not set in the Bit 3, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. Writing a 1 to this bit (in any of the controllers) will clear the bit. The A20Gate Pass-Through Logic allows specific port 64h Writes to complete without setting this bit. 0 = No event occurred. 1 = Event Occurred. ***Note:*** Software clears this bit by writing a 1 to the bit location in any of the controllers		0h	RWC
10	TRAPBY64R	**SMI Caused by Port 64 Read:** Indicates if the event occurs. Even if the corresponding enable bit is not set in the Bit 2, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. Writing a 1 to this bit (in any of the controllers) will clear the bit. 0 = No event occurred. 1 = Event Occurred. ***Note:*** Software clears this bit by writing a 1 to the bit location in any of the controllers		0h	RWC

Table 25-16. USBLKMCR - USB Legacy Keyboard / Mouse Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** C0h **Offset End:** C1h	
Size: 16 bit	**Default:** 2000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
09	TRAPBY60W	**SMI Caused by Port 60 Write:** Indicates if the event occurs. Even if the corresponding enable bit is not set in the Bit 1, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. Writing a 1 to this bit (in any of the controllers) will clear the latch. The A20Gate Pass-Through Logic allows specific port 60h Writes to complete without setting this bit.		0h	RWC
08	TRAPBY60R	**SMI Caused by Port 60 Read:** Indicates if the event occurs. Even if the corresponding enable bit is not set in the Bit 0, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#. Writing a 1 to this bit (in any of the controllers) will clear the latch.		0h	RWC
07	SMIATENDPS	**SMI at End of Pass-through Enable:** This bit enables SMI at the end of a pass-through. This can occur if an SMI is generated in the middle of a pass through, and needs to be serviced later. 0 = Disable 1 = Enable		0h	RW
06	PSTATE	**Pass Through State:** 0 = If software needs to reset this bit, it must set bit 5 in all of the host controllers to 0. 1 = Indicates that the state machine is in the middle of an A20GATE pass-through sequence. ***Note:*** Software must set bit 5 in all of the host controllers to 0 to reset this bit.		0h	RO
05	A20PASSEN	**A20Gate Pass-Through Enable:** 0 = Disable. 1 = Enable. Allows A20GATE sequence Pass-Through function. A specific cycle sequence involving writes to port 60h and 64h does not result in the setting of the SMI status bits. SMI# will not be generated, even if the various enable bits are set.		0h	RW
04	USBSMIEN	**SMI on USB IRQ:** 0 = Disable. The USB interrupt will not cause an SMI event. 1 = Enable. The USB interrupt will cause an SMI event.		0h	RW
03	64WEN	**SMI on Port 64 Writes Enable:** 0 = Disable. A 1 in bit 11 will not cause an SMI event. 1 = Enable. A 1 in bit 11 will cause an SMI event.		0h	RW
02	64REN	**SMI on Port 64 Reads Enable:** 0 = Disable. A 1 in bit 10 will not cause an SMI event. 1 = Enable. A 1 in bit 10 will cause an SMI event.		0h	RW
01	60WEN	**SMI on Port 60 Writes Enable:** 0 = Disable. A 1 in bit 9 will not cause an SMI event. 1 = Enable. A 1 in bit 9 will cause an SMI event.		0h	RW
00	60REN	**SMI on Port 60 Reads Enable:** 0 = Disable. A 1 in bit 8 will not cause an SMI event. 1 = Enable. A 1 in bit 8 will cause an SMI event.		0h	RW

25.1.1.16 USBREN - USB Resume Enable Register

Note: There is no support for wake from USB when in S3/S4/S5.

Table 25-17. USBREN - USB Resume Enable Register

Description:					
Size: 8 bit	**Default:** 00h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		0h	
01	PORT1EN	0 = The USB controller will not look at this port for a wakeup event. 1 = Enables port 1 of the USB controller to look at wakeup events. When set, the USB controller will monitor port 1 for remote wakeup and connect/disconnect events.		0h	RW
00	PORT0EN	0 = The USB controller will not look at this port for a wakeup event. 1 = Enables port 0 of the USB controller to look at wakeup events. When set, the USB controller will monitor port 0 for remote wakeup and connect/disconnect events.		0h	RW

25.1.1.17 USBCWP - USB Core Well Policy Register

Table 25-18. USBCWP - USB Core Well Policy Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** C8h **Offset End:** C8h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved	Reserved		0h	
00	SBMSPEN	**Static Bus Master Status Policy Enable:** 0 = The UHCI Host Controller dynamically sets the Bus Master status bit based on the memory accesses that are scheduled. See Section 25.5 for details. 1 = The UHCI host controller statically forces the Bus Master Status bit in power management space to 1 whenever the HCHalted bit is cleared.		0h	RW

25.1.1.18 MANID - Manufacturer ID Register

Table 25-19. MANID - Manufacturer ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:0	**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:**	00010F90h		**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		00h	RO
23 : 16	SID	**Stepping ID:** This field increments for each stepping of the part. This field can be used by software to differentiate steppings when the Revision ID may not change. ***Note:*** 00h for A0 stepping ***Note:*** 01h for B0 stepping		01h	RO
15 : 08	MID	**Manufacturer:** 0Fh = Intel		0Fh	RO
07 : 00	Reserved	Reserved1263		90h	

25.2 USB (1.1) Controller I/O-Mapped Register Details

Some of the read/write register bits which deal with changing the state of the USB hub ports function such that on read back they reflect the current state of the port and not necessarily the state of the last write to the register. This allows the software to poll the state of the port and wait until it is in the proper state before proceeding. A Host controller reset, global reset, or port reset will immediately terminate a transfer on the affected ports and disable the port. This affects the USBCMD register, bit [4] and the PORTSC registers, bits [12,6,2]. See individual bit descriptions for more detail.

Base address for these I/O registers is set by the USBIOBAR register, see Section 25.1.1.9, “USBIOBAR - Base Address Register” on page 946.

Table 25-20. Summary of USB (1.1) Controller Configuration Registers Mapped Through USBIOBAR I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	“USBCMD: USB Command Register” on page 954	0000h
02h	03h	“USBSTS: USB Status Register” on page 957	0020h
04h	05h	“USBINTR: USB Interrupt Enable Register” on page 959	0000h
06h	07h	“FRNUM: Frame Number Register” on page 959	0000h
08h	0Bh	“FRBASEADD: Frame List Base Address Register” on page 960	XXXXX000h
0Ch	0Ch	“SOFMOD: Start of Frame Modify Register” on page 961	40h
10h	11h	“PSCR - Port Status and Control Register” on page 962	0080h
12h	13h	“PSCR - Port Status and Control Register” on page 962	0080h

25.2.1 Register Details

25.2.1.1 USBCMD: USB Command Register

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed. Table 25-22 provides additional information on the operation of the Run/Stop and Debug bits.

Table 25-21. USBCMD: USB Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved.		0h	
08	Reserved	1 = Reserved		0h	RW
07	MAXP	**Max Packet:** 0 = 32 bytes. 1 = 64 bytes. This bit selects the maximum packet size that can be used for full speed bandwidth reclamation at the end of a frame. This value is used by the Host Controller to determine whether it should initiate another transaction based on the time remaining in the SOF counter. Use of reclamation packets larger than the programmed size will cause a Babble error if executed during the critical window at frame end. The Babble error results in the offending endpoint being stalled. Software is responsible for ensuring that any packet which could be executed under bandwidth reclamation be within this size limit.		0h	RW
06	CF	**Configure Flag:** 0 = Indicates that software has not completed host controller configuration. 1 = HCD software sets this bit as the last action in its process of configuring the host controller. This bit has no effect on the hardware. It is provided only as a semaphore service for software.		0h	RW
05	SWDBG	**Software Debug:** 0 = Normal Mode. 1 = Debug mode. In software Debug mode, the Host Controller clears the Run/Stop bit after the completion of each USB transaction. The next transaction is executed when software sets the Run/Stop bit back to 1. The SWDBG bit must only be manipulated when the controller is in the stopped state. This can be determined by checking the HCHalted bit in the USBSTS register.		0h	RW
04	FGR	**Force Global Resume:** 0 = Software sets this bit to 0 after 20 ms has elapsed to stop sending the Global Resume signal. At that time all USB devices must be ready for bus activity. 1 = Host Controller sends the Global Resume signal on the USB. The Host Controller sets this bit to 1 when a resume event (connect, disconnect, or K-state) is detected while in global suspend mode. The 1 to 0 transition causes the port to send a low speed EOP signal. This bit will remain a 1 until the EOP has completed.		0h	RW

Table 25-21. USBCMD: USB Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	EGSM	**Enter Global Suspend Mode:** 0 = Software resets this bit to 0 to come out of Global Suspend mode. Software writes this bit to 0 at the same time that Force Global Resume (bit 4) is written to 0 or after writing bit 4 to 0. 1 = Host Controller enters the Global Suspend mode. No USB transactions occurs during this time. The Host Controller is able to receive resume signals from USB and interrupt the system. Software must ensure that the Run/Stop bit (bit 0) is cleared prior to setting this bit.		0h	RW
02	GRESET	**Global Reset:** 0 = This bit is reset by the software after a minimum of 10 ms has elapsed as specified in the *Universal Serial Bus (USB) Specification, Rev. 2.0*. 1 = Global Reset. The host controller sends the global reset signal on the USB and then resets all its logic. Chip Hardware Reset has the same effect as Global Reset (bit 2), except that the host controller does not send the Global Reset on USB.		0h	RW
01	HCRESET	**Host Controller Reset:** The HCReset effects on hub registers are slightly different from Chip Hardware Reset and Global USB Reset. The HCReset affects bits [8,3:0] of the Port Status and Control Register (PORTSC) of each port. HCReset resets the state machines of the Host Controller including the Connect/Disconnect state machine (one for each port). When the Connect/ Disconnect state machine is reset, the output that signals connect/disconnect are negated to 0, effectively signaling a disconnect, even if a device is attached to the port. This virtual disconnect causes the port to be disabled. This disconnect and disabling of the port causes bit 1 (connect status change) and bit 3 (port enable/disable change) of the PORTSC to get set. The disconnect also causes bit 8 of PORTSC to reset. About 64 bit times after HCReset goes to 0, the connect and low-speed detect will take place and bits 0 and 8 of the PORTSC will change accordingly. 0 = Reset by the host controller when the reset process is complete. 1 = Reset. When this bit is set, the host controller module resets its internal timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated.		0h	RW
00	RS	**Run / Stop:** 0 = Stop. Completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. The Host Controller clears this bit when the following fatal errors occur: consistency check failure, Memory access errors. 1 = Run. Proceeds with execution of the schedule and continues execution as long as this bit is set.		0h	RW

Table 25-22. Run/Stop, Debug Bit Interaction SWDBG (Bit 5), Run/Stop (Bit 0) Operation

SWDBG (Bit 5)	Run/ Stop (Bit 0)	Description
0	0	If executing a command, the Host Controller completes the command and then stops. The 1.0 ms frame counter is reset and command list execution resumes from start of frame using the frame list pointer selected by the current value in the FRNUM register. (While Run/Stop=0, the FRNUM register can be reprogrammed).
0	1	Execution of the command list resumes from Start Of Frame using the frame list pointer selected by the current value in the FRNUM register. The Host Controller remains running until the Run/Stop bit is cleared (by Software or Hardware).
1	0	If executing a command, the Host Controller completes the command and then stops and the 1.0 ms frame counter is frozen at its current value. All status are preserved. The Host Controller begins execution of the command list from where it left off when the Run/Stop bit is set.
1	1	Execution of the command list resumes from where the previous execution stopped. The Run/Stop bit is set to 0 by the Host Controller when a TD is being fetched. This causes the Host Controller to stop again after the execution of the TD (single step). When the Host Controller has completed execution, the HC Halted bit in the Status Register is set.

When the USB Host Controller is in Software Debug Mode (USBCMD Register bit 5=1), the single stepping software debug operation is as follows:

To Enter Software Debug Mode:

1. HCD puts Host Controller in Stop state by setting the Run/Stop bit to 0.
2. HCD puts Host Controller in Debug Mode by setting the SWDBG bit to 1.
3. HCD sets up the correct command list and Start Of Frame value for starting point in the Frame List Single Step Loop:
4. HCD sets Run/Stop bit to 1.
5. Host Controller executes next active TD, sets Run/Stop bit to 0, and stops.
6. HCD reads the USBCMD register to check if the single step execution is completed (HCHalted=1).
7. HCD checks results of TD execution. Go to step 4 to execute next TD or step 8 to end Software Debug mode.
8. HCD ends Software Debug mode by setting SWDBG bit to 0.
9. HCD sets up normal command list and Frame List table.
10. HCD sets Run/Stop bit to 1 to resume normal schedule execution.

In Software Debug mode, when the Run/Stop bit is set, the Host Controller starts. When a valid TD is found, the Run/Stop bit is reset. When the TD is finished, the HCHalted bit in the USBSTS register (bit 5) is set.

The software Debug mode skips over inactive TDs and only halts after an active TD has been executed. When the last active TD in a frame has been executed, the Host Controller waits until the next SOF is sent and then fetches the first TD of the next frame before halting.

This HCHalted bit can also be used outside of Software Debug mode to indicate when the Host Controller has detected the Run/Stop bit and has completed the current transaction. Outside of the Software Debug mode, setting the Run/Stop bit to 0 always resets the SOF counter so that when the Run/Stop bit is set the Host Controller starts over again from the frame list location pointed to by the Frame List Index (see FRNUM Register description) rather than continuing where it stopped.

25.2.1.2 USBSTS: USB Status Register

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it. See Section 4, “Interrupts,” of the *Universal Host Controller Interface (UHCI) Specification, Rev. 1.1*, for additional information concerning USB interrupt conditions.

Table 25-23. USBSTS: USB Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 0020h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 06	Reserved	Reserved		0h	RO
05	HCH	**HCHalted:** 0 = The host controller is running. 1 = The host controller has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the host controller hardware (debug mode or an internal error). Default. Software clears this bit by writing a 1 to it.		1	RWC
04	HCPE	**Host Controller Process Error:** 0 = No error. 1 = The host controller has detected a fatal error. This indicates that the Host Controller suffered a consistency check failure while processing a Transfer Descriptor. An example of a consistency check failure would be finding an illegal PID field while processing the packet header portion of the TD. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further schedule execution. A hardware interrupt is generated to the system. Software clears this bit by writing a 1 to it.		0h	RWC
03	HSE	**Host System Error:** 0 = No error occurred. 1 = A serious error occurs during a host system access involving the Host Controller module. Conditions that set this bit to 1 include Parity error, Master Abort, and Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. A hardware interrupt is generated to the system. Software clears this bit by writing a 1 to it.		0h	RWC

Table 25-23. USBSTS: USB Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 0020h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	RSM_DET	**Resume Detect:** 0 = Resume not detected. 1 = The host controller received a "RESUME" signal from a USB device. This is only valid if the Host controller is in a global suspend state (bit 3 of Command register = 1). Software clears this bit by writing a 1 to it.		0h	RWC
01	USBEINT	**USB Error Interrupt:** The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and Bit 0 are set. 0 = No **USB Error Interrupt.** 1 = Completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and Bit 0 are set. Software clears this bit by writing a 1 to it.		0h	RWC
00	USBINT	**USB Interrupt:** 0 = No USB interrupt. 1 = The Host Controller sets this bit to 1 when the cause of an interrupt is a completion of a USB transaction whose Transfer Descriptor had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual length field in TD is less than maximum length field in TD), and short packet detection is enabled in that TD. Software clears this bit by writing a 1 to it.		0h	RWC

25.2.1.3 USBINTR: USB Interrupt Enable Register

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Fatal errors (Host Controller Processor Error-bit 4, USBSTS Register) cannot be disabled by the host controller. Interrupt sources that are disabled in this register still appear in the Status Register to allow the software to poll for events.

Table 25-24. USBINTR: USB Interrupt Enable Register

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 05	Reserved	Reserved		0h	
04	Scratchpad	Scratchpad		0h	RW
03	SPIEN	**Short Packet Interrupt Enable:** 0 = Disabled 1 = Enabled.		0h	RW
02	IOC	**Interrupt On Complete:** 0 = Disabled 1 = Enabled.		0h	RW
01	RIEN	**Resume Interrupt Enable:** 0 = Disabled 1 = Enabled.		0h	RW
00	TIEN	**Timeout/CRC Interrupt Enable:** 0 = Disabled 1 = Enabled.		0h	RW

25.2.1.4 FRNUM: Frame Number Register

Bits [10:0] of this register contain the current frame number which is included in the frame SOF packet. This register reflects the count value of the internal frame number counter. Bits [9:0] are used to select a particular entry in the Frame List during schedule execution. This register is updated at the end of each frame time.

This register must be written as a word. Byte writes are not supported. This register cannot be written unless the Host Controller is in the STOPPED state as indicated by the HCHalted bit (USBSTS register). A write to this register while the Run/Stop bit is set (USBCMD register) is ignored.

Table 25-25. FRNUM: Frame Number Register

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RO
10 : 00	FLCIFN	**Frame List Current Index/Frame Number:** Bits [10:0] provide the frame number in the SOF Frame. The value in this register increments at the end of each time frame (approximately every 1 ms). In addition, bits [9:0] are used for the Frame List current index and correspond to memory address signals [11:2].		0h	RW

25.2.1.5 FRBASEADD: Frame List Base Address Register

This 32-bit register contains the beginning address of the Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. When written, only the upper 20 bits are used. The lower 12 bits are written as zero (4 Kbyte alignment). The contents of this register are combined with the frame number counter to enable the Host Controller to step through the Frame List in sequence. The two least significant bits are always 00. This requires DWord alignment for all list entries. This configuration supports 1024 Frame List entries.

Table 25-26. FRBASEADD: Frame List Base Address Register

Description:					
View: PCI	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** XXXXX000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	BAD	**Base Address:** These bits correspond to memory address signals [31:12], respectively.		X	RW
11 : 00	Reserved	Reserved. Must be written as 0s.		0h	

25.2.1.6 SOFMOD: Start of Frame Modify Register

This 1-byte register is used to modify the value used in the generation of SOF timing on the USB. Only the 7 least significant bits are used. When a new value is written into the these 7 bits, the SOF timing of the next frame will be adjusted. This feature can be used to adjust out any offset from the clock source that generates the clock that drives the SOF counter. This register can also be used to maintain real time synchronization with the rest of the system so that all devices have the same sense of real time. Using this register, the frame length can be adjusted across the full range required by the *USB Specification*. Its initial programmed value is system dependent based on the accuracy of hardware USB clock and is initialized by system BIOS. It may be reprogrammed by USB system software at any time. Its value will take effect from the beginning of the next frame. This register is reset upon a Host Controller Reset or Global Reset. Software must maintain a copy of its value for reprogramming if necessary.

Table 25-27. SOFMOD: Start of Frame Modify Register

Description:					
View: PCI	**BAR:** USBIOBAR (IO)		**Bus:Device:Function:** 0:29:0	**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:** 40h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0h	
06 : 00	SOFTV	**SOF Timing Value:** Guidelines for the modification of frame time are contained in Chapter 7 of the *USB Specification*. The SOF cycle time (number of SOF counter clock periods to generate a SOF frame length) is equal to 11936 + value in this field. The default value is decimal 64 which gives a SOF cycle time of 12000. For a 12 MHz SOF counter clock input, this produces a 1 ms Frame period. The following table indicates what SOF Timing Value to program into this field for a certain frame period. **Frame Length (# 12 MHz Clocks) (decimal)** / **SOF Timing Value (this register) (decimal)** 11936 / 0 11937 / 1 — / — 11999 / 63 12000 / 64 12001 / 65 — / — 12062 / 126 12063 / 127		40h	RW

25.2.1.7 PSCR - Port Status and Control Register

After a Power-up reset, Global reset, or Host Controller reset, the initial conditions of a port are: No device connected, Port disabled, and the bus line status is 00 (single-ended zero).

Table 25-28. PSCR - Port Status and Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:**[a] 10h **Offset End:** 11h	
View: PCI 2	**BAR:** USBIOBAR (IO)	**Bus:Device:Function:** 0:29:0		**Offset Start:** 12h **Offset End:** 13h	
Size: 16 bit	**Default:** 0080h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 13	Reserved	Reserved		0h	
12	SUS	**Suspend:** 0 = Port not in suspend state. 1 = Port in suspend state. This bit should not be written to a 1 if global suspend is active (bit 3=1 in the USBCMD register). This bit and bit 2 below define the hub states: **Bits [12,2]** **Hub State** X,0 Disable 0,1 Enable 1,1 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for single-ended 0 resets (global reset and port reset). The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. The bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. Normally, if a transaction is in progress when this bit is set, the port will be suspended when the current transaction completes. However, in the case of a specific error condition (out transaction with babble), the CMI may issue a start-of-frame, and then suspend the port.		0h	RW
11	OCI	**Overcurrent Indicator:** This bit is cleared by software writing a '1'. 0 = Overcurrent pin inactive. 1 = Overcurrent pin has gone from inactive to active on this port. Software clears this bit by writing a 1 to it.		0h	RWC
10	OCS	**Overcurrent Status:** This bit indicates the current status of the OC# pin for this port. This bit is set and cleared by hardware. 0 = This port does not have an overcurrent condition. 1 = This port currently has an overcurrent condition.		0h	RO
09	PRST	**Port Reset:** 0 = Port is not disabled. 1 = Port is disabled and sends the USB Reset signaling.		0h	RW
08	LS	**Low Speed Device Attached:** 0 = Full speed device is attached. 1 = Low speed device is attached to this port. Writes have no effect.		0h	RO
07	Reserved	Reserved: Always read as 1.		1	

Table 25-28. PSCR - Port Status and Control Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** USBIOBAR (IO)		**Bus:Device:Function:** 0:29:0	**Offset Start:**[a] 10h **Offset End:** 11h	
View: PCI 2	**BAR:** USBIOBAR (IO)		**Bus:Device:Function:** 0:29:0	**Offset Start:** 12h **Offset End:** 13h	
Size: 16 bit	**Default:** 0080h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	RSM_DET	**Resume Detect:** 0 = No resume (K-state) detected/driven on port. 1 = Resume detected/driven on port. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected for at least 32 µs while the port is in the Suspend state. CMI will then reflect the K-state back onto the bus as long as the bit remains a '1', and the port is still in the suspend state (bit 12,2 are '11') Writing a 0 (from 1) causes the port to send a low speed EOP. This bit will remain a 1 until the EOP has completed.		0h	RW
05 : 04	LNS	**Line Status:** These bits reflect the D+ (bit 4) and D- (bit 5) signals lines logical levels. These bits are used for fault detect and recovery as well as for USB diagnostics. This field is updated at EOF2 time.		00h	RO
03	PORT_ENC	**Port Enable / Disable Change:** 0 = No change. 1 = Port enabled/disabled status has changed. For the root hub, this bit gets set only when a port is disabled due to disconnect on the that port or due to the appropriate conditions existing at the EOF2 point (See the *USB Specification*). Software clears this bit by writing a 1 to it.		0h	RWC
02	PORT_EN	**Port Enabled / Disabled:** 0 = Disable. 1 = Enable. Ports can be enabled by host software only. Ports can be disabled by either a fault condition (disconnect event, overcurrent, or other fault condition) or by host software. The bit status does not change until the port state actually changes and that there may be a delay in disabling or enabling a port if there is a transaction currently in progress on the USB.		0h	RW
01	CSC	**Connect Status Change:** 0 = No change. 1 = Change in Current Connect Status. Indicates a change has occurred in the port's Current Connect Status (see bit 0). The hub device sets this bit for any changes to the port device connect status, even if system software has not cleared a connect status change. If, for example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit will remain set). However, the hub transfers the change bit only once when the Host Controller requests a data transfer to the Status Change endpoint. System software is responsible for determining state change history in such a case. Software sets this bit to 0 by writing a 1 to it.		0h	RWC
00	CCS	**Current Connect Status:** 0 = No device is present. 1 = Device is present on port. This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.		0h	RO

a. USB #1 Port 0: 10-11hPort 1: 12-13h

25.3 Data Transfers to/from Main Memory

The *Universal Host Controller Interface (UHCI) Specification, Rev. 1.1* describes the details on how HCD and CMI communicate via the Schedule data structures.

25.4 Data Structures in Main Memory

The *UHCI Specification* details the data structures used to communicate control, status, and data between software and CMI.

25.5 Data Transfers To/From Main Memory

The following sections describe the details on how HCD and CMI communicate via the Schedule data structures. The discussion is organized in a top-down manner, beginning with the basics of walking the Frame List, followed by a description of generic processing steps common to all transfer descriptors, and finally a discussion on Transfer Queuing.

During data transfers to and from main memory, the UHCI DMA engine must provide an indication to the processor power management logic that it is busy. Therefore, the memory accesses may actually be “cache accesses”. The indication to the power management logic will be referred to as “UHCI Bus Master Status”.

The UHCI controllers offer two different policies for the generation of the UHCI Bus Master Status: static and dynamic.

The static policy requires that the UHCI Bus Master Status signal is asserted when the HCHalted bit is 0. This policy prevents C3/C4 entry whenever the DMA engine is enabled.

The dynamic policy requires that the UHCI Bus Master Status signal deasserts after completely traversing the schedule for a particular Frame unless the Frame bit counter indicates that the next Frame begins in less than ~100 usecs. Specifically the UHCI Bus Master Status signal must deassert in any of the following:

1. After reading the Frame List Pointer and detecting that the Terminate bit is set and the next Frame is greater than ~100 usecs in the future.
2. After servicing a Queue Head in which the Terminate bit is set in the Queue Head Link Pointer and the next Frame is greater than ~100 usecs in the future.
3. The RUN bit is 0 and the HCHalted bit is 1
4. After servicing a Transfer Descriptor which is not in the context of a Queue and it's Terminate bit is set and the next Frame is greater than ~100 us in the future

The UHCI Bus Master Status signal must assert in either of the following cases:

1. The RUN bit transitions from 0 to 1
2. The Frame bit counter indicates that the next Frame begins in less than ~100 µs

The value of 100 µs is somewhat arbitrary. This is based on the assumption that a “reasonable worst case” exit time is approximately 100 µs. Even if the exit latency is greater than this, it does not mean that there will be a USB functional failure. It only means that the USB traffic will begin later in the upcoming Frame than it normally would have. In order to specify numbers for checking in validation, the ~100 µs time must fall within the range 97 µs to 110 µs.

The UHCI BM_STS Static Policy Enable configuration bit (D31.F0.A9h.5) selects between the dynamic and static policies.

25.5.1 Executing the Schedule

Software programs the starting address of the Frame List and the Frame List index, then causes CMI to execute the schedule by setting the Run/Stop bit in the Control register to Run. The CMI processes the schedule one entry at a time: the next element in the frame list is not fetched until the current element in the frame list is retired.

Schedule execution proceeds in the following fashion:

- The CMI first fetches an entry from the Frame List. This entry has three fields. Bit 0 indicates whether the address pointer field is valid. Bit 1 indicates whether the address points to a Transfer Descriptor or to a queue head. The third field is the pointer itself.
- If the Frame List entry indicates that it points to a Transfer Descriptor, the CMI fetches the entry and begins the operations necessary to initiate a transaction on USB. Each TD contains a link field that points to the next entry, as well as indicating whether it is a TD or a QH.
- If the Frame List entry contains a pointer to a QH, the CMI processes the information from the QH to determine the address of the next data object that it must process.
- The TD/QH process continues until the millisecond allotted to the current frame expires. At this point, the CMI fetches the next entry from the Frame List. If the CMI is not able to process all of the transfer descriptors during a given frame, those descriptors are retired by software without having been executed.

25.5.2 Processing Transfer Descriptors

The CMI executes a TD using the following, generalized algorithm. These basic steps are common across all modes of TDs. Subsequent sections present processing steps unique to each TD mode.

1. The CMI Fetches TD or QH from the current Link Pointer.
2. If a QH, go to 1 to fetch from the Queue Element Link Pointer. If inactive, go to 12
3. Build Token, actual bits are in TD Token.
4. If (Host-to-Function) then
   [*Memory Access*] issue request for data, (referenced through TD.BufferPointer)
   wait for first chunk data arrival
   end if
5. [*Begin USB Transaction*] Issue Token (from token built in 2, above) and begin data transfer.
   if (Host-to-Function) then Go to 6
   else Go to 7
   end if
6. Fetch data from memory (via TD BufferPointer) and transfer over USB until TD Maximum-Length bytes have been read and transferred. [*Concurrent system memory and USB Accesses*]. Go to 8.
7. Wait for data to arrive (from USB). Write incoming bytes into memory beginning at TD BufferPointer. Internal HC buffer must signal end of data packet. Number of bytes received must be ≤ TD Maximum-Length; The length of the memory area referenced by TD BufferPointer must be ≥ TD Maximum-Length. [*Concurrent system memory and USB Accesses*].
8. Issue handshake based on status of data received (Ack or Time-out). Go to 10.
9. Wait for handshake, if required [*End of USB Transaction*].
10. Update Status [*Memory Access*] (TD.Status and TD.ActualLength).
    If the TD completed successfully, mark the TD inactive. Go to 11.

If not successful, and the error count has not been reached, leave the TD active. If the error count has been reached, mark the TD inactive. Go to 12.

11. Write the link pointer from the current TD into the element pointer field of the QH structure. If the Vf bit is set in the link pointer, go to 2.
12. Proceed to next entry.

25.5.3 Command Register, Status Register, and TD Status Bit Interaction

Condition	USB Status Register Actions	TD Status Register Actions
CRC/Time Out Error	Set USB Error Int bit[1], Clear HC Halted bit	Clear Active bit[1] and set Stall bit[1]
Illegal PID, PID Error, Maximum Length (illegal)	Clear Run/Stop bit in command register Set HC Process Error and HC Halted bits	
Master/Target Abort on memory accesses	Clear Run/Stop bit in command register Set Host System Error and HC Halted bits	
Suspend Mode	Clear Run/Stop bit in command register[2] Set HC Halted bit	
Resume Received Suspend Mode = 1	Set Resume received bit	
Run/Stop = 0	Clear Run/Stop bit in command register Set HC Halted bit	
Configuration Flag Set	Set Configuration Flag in command register	
HC Reset/Global Reset	Clear Run/Stop and configuration Flag in command register Clear USB Int, USB Error Int, Resume received, Host System Error, HC Process Error, and HC Halted bits	
IOC = 1 in TD Status	Set USB Int bit	
Stall	Set USB Error Int bit	Clear Active bit[1] and set Stall bit
Bit Stuff/Data Buffer Error	Set USB Error Int bit[1]	Clear Active bit[1] and set Stall bit[1]
Short Packet Detect	Set USB Int bit	Clear Active bit

Notes:
1. Only If error counter counted down from 1 to 0.
2. Suspend mode can be entered only when Run/Stop bit is 0.

Note: If a NAK or STALL response is received from a SETUP transaction, a Time Out Error will be reported. This will cause the Error counter to decrement and the CRC/Time-out Error status bit to be set within the TD Control and Status Dword during write back. If the Error counter changes from 1 to 0, the Active bit will be reset to 0 and Stalled bit to 1 as normal.

25.5.4 Transfer Queuing

Transfer Queues are used to implement a guaranteed data delivery stream to a USB Endpoint. Transfer Queues are composed of two parts: a Queue Header (QH) and a linked list. The linked list of TDs and QHs has an indeterminate length (0 to n).

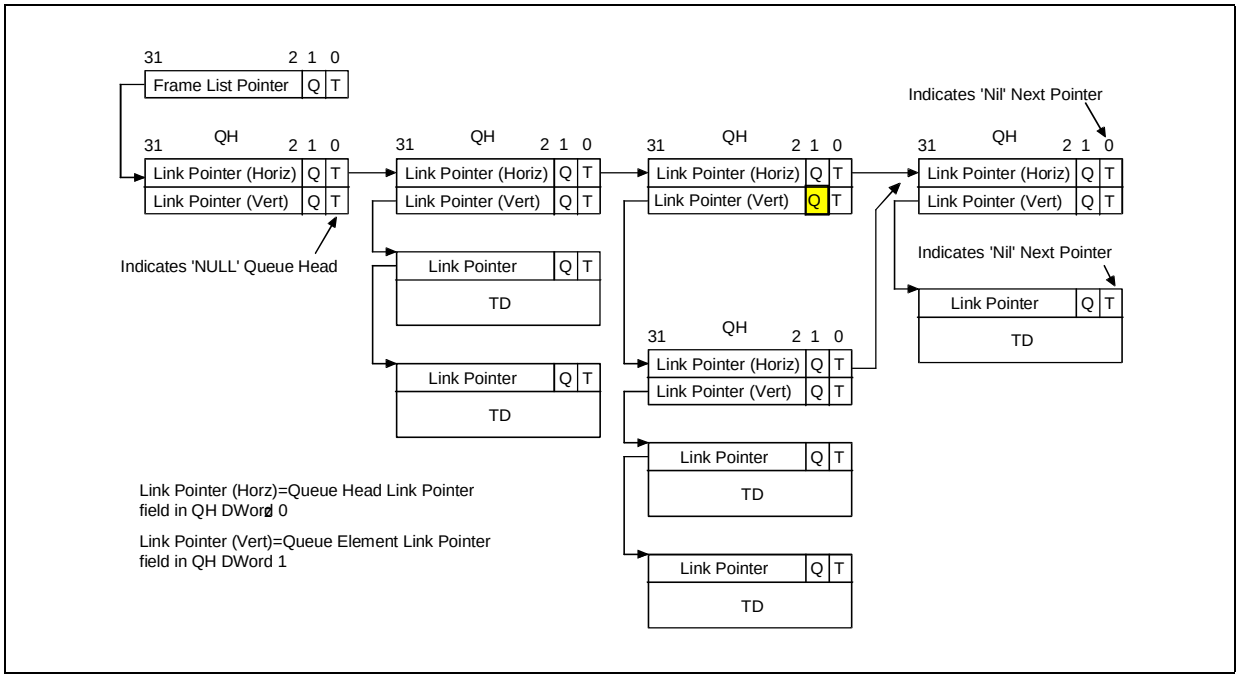
The QH contains two link pointers and is organized as two contiguous DWords. The first DWord is a horizontal pointer (Queue Head Link Pointer), used to link a single transfer queue with either another transfer queue, or a TD (target data structure depends on Q bit). If the T bit is set, this QH represents the last data structure in the current Frame. The T bit informs the CMI that no further processing is required until the beginning of the next frame. The second DWord is a vertical pointer (Queue Element Link Pointer) to the first data structure (TD or QH) being managed by this QH. If the T bit is set, the queue is empty. This pointer may reference a TD or another QH.

Figure 25-1 illustrates four example queue conditions. The first QH (on far left) is an example of an “empty” queue; the termination bit (T Bit), in the vertical link pointer field, is set to 1. The horizontal link pointer references another QH. The next queue is the expected typical configuration. The horizontal link pointer references another QH, and the vertical link pointer references a valid TD.

Typically, the vertical pointer in a QH points to a TD. However, as shown in Figure 25-1 (third example from left side of figure) the vertical pointer could point to another QH. When this occurs, a new Q Context is entered and the Q Context just exited is NULL (The CMI will not update the vertical pointer field).

The far right QH is an example of a frame ‘termination’ node. Since its horizontal link pointer has its termination bit set, the CMI assumes there is no more work to complete for the current Frame.

Figure 25-1. Example Queue Conditions



Transfer Queues are based on the following characteristics:

- A QH’s vertical link pointer (Queue Element Link Pointer) references the ‘Top’ queue member. A QH’s horizontal link pointer (Queue Head Link Pointer) references the “next” work element in the Frame.
- Each queue member’s link pointer references the next element within the queue.

In the simplest model, the CMI follows vertical link point to a queue element, then executes the element. If the completion status of the TD satisfies the advance criteria as shown in Table 25-29, the CMI advances the queue by writing the just-executed TD’s link pointer back into the QH’s Queue Element link pointer. The next time the queue head is traversed, the next queue element will be the Top element.

The traversal has two options: Breadth first, or Depth first. A flag bit in each TD (Vf - Vertical Traversal Flag) controls whether traversal is Breadth or Depth first. The default mode of traversal is Breadth-First. For Breadth-First, the CMI only executes the top element from each queue. The execution path is:

QH (Queue Element Link Pointer)→ TD

→ Write-Back to QH (Queue Element Link Pointer)

→ QH (Queue Head Link pointer).

Breadth-First is also performed for every transaction execution that fails the advance criteria. This means that if a queued TD fails, the queue does not advance, and the CMI traverses the QH's Queue Head Link Pointer.

In a Depth-first traversal, the top queue element must complete successfully to satisfy the *advance criteria* for the queue. If the CMI is currently processing a queue, and the advance criteria are met, and the Vf bit is set, the CMI follows the TD's link pointer to the next schedule work item.

Regardless of traversal model, when the advance criteria are met, the successful TD's link pointer is written back to the QH's Queue Element link pointer.

When the CMI encounters a QH, it caches the QH internally, and sets internal state to indicate it is in a Q-context. It needs this state to update the correct QH (for auto advancement) and also to make the correct decisions on how to traverse the Frame List.

Restricting the advancement of queues to advancement criteria implements a guaranteed data delivery stream.

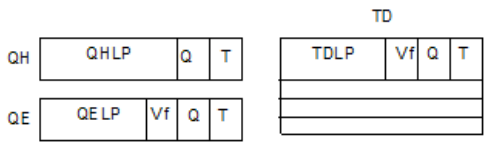
A queue is **never** advanced on an error completion status (even in the event the error count was exhausted).

Table 25-29 lists the general queue advance criteria, which are based on the execution status of the TD at the 'Top' of a currently 'active' queue.

Table 25-29. Queue Advance Criteria

Function-to-Host (IN)			Host-to-Function (OUT)		
Non-NULL	**NULL**	**Error/NAK**	**Non-NULL**	**NULL**	**Error/NAK**
Advance Q	Advance Q	Retry Q Element	Advance Q	Advance Q	Retry Q Element

Table 25-30 is a decision table illustrating the valid combinations of link pointer bits and the valid actions taken when advancement criteria for a queued transfer descriptor are met. The column headings for the link pointer fields are encoded, based on the following list:



Legend:
QH.LP = Queue Head Link Pointer (or Horizontal Link Pointer)QE.Q = Q bit in QE
QE.LP = Queue Element Link Pointer (or Vertical Link Pointer)QE.T = T bit in QE
TD.LP = TD Link PointerTD. Vf = Vf bit in TD
QH.Q = Q bit in QH TD.Q = Q bit in TD
QH.T = T bit in QH TD. Vf = Vf bit in TD

Table 25-30. USB Schedule List Traversal Decision Table

Q Context	QH.Q	QH.T	QE.Q	QE.T	TD.Vf	TD.Q	TD.T	Description
0	-	-	-	-	x	0	0	Not in Queue – execute TD. Use TD.LP to get next TD
0	-	-	-	-	x	x	1	Not in Queue – execute TD. End of Frame
0	-	-	-	-	x	1	0	Not in Queue - execute TD. Use TD.LP to get next (QH+QE). Set Q Context to 1.
1	0	0	0	0	0	x	x	In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use QH.LP to get next TD.
1	x	X	0	0	1	0	0	In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use TD.LP to get next TD.
1	x	X	0	0	1	1	0	In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use TD.LP to get next (QH+QE).
1	0	0	x	1	x	x	x	In Queue. Empty queue. Use QH.LP to get next TD
1	x	X	1	0	-	-	-	In Queue. Use QE.LP to get (QH+QE)
1	x	1	0	0	0	x	x	In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. End of Frame
1	x	1	x	1	x	x	x	In Queue. Empty queue. End of Frame
1	1	0	0	0	0	x	x	In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use QH.LP to get next (QH+QE).
1	1	0	x	1	x	x	x	In Queue. Empty queue. Use QH.LP to get next (QH+QE)

25.6 USB Buffer Management

The USB controller contains a 64 byte FIFO, which operate in a ping/pong fashion. This buffer is not aligned to any memory boundary. Upon each new transfer descriptor (TD), the buffer resets its pointer to the top.

When transmitting to a USB port, if the length field from the TD is greater than 32 bytes, meaning the internal FIFO boundary will be crossed, the host controller will read both ping and pong together. When ping has been transferred to USB, and the total TD length has not been fetched, the USB host controller will fetch another 32 bytes.

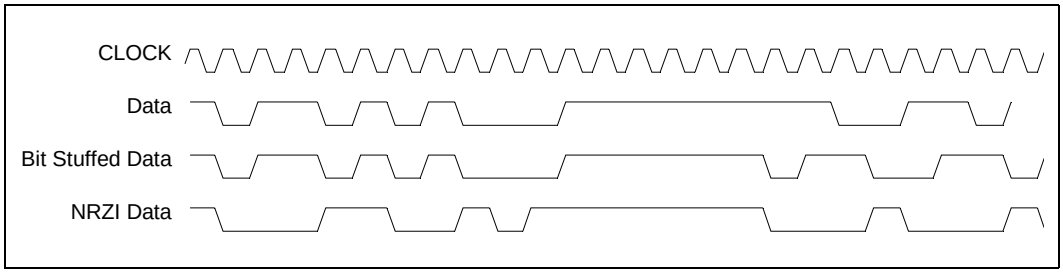
When receiving from a USB port, it is not known how many bytes will be received. Therefore, the CMI will flush each buffer to memory as it becomes full.

25.7 Data Encoding and Bit Stuffing

The CMI USB employs NRZI data encoding (Non-Return to Zero Inverted) when transmitting packets. Full details on this implementation are given in the *USB Specification, Rev. 2.0*.

The USB employs NRZI data encoding when transmitting packets. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. A string of zeros causes the NRZI data to toggle each bit time. A string of ones causes long periods with no transitions in the data. In order to ensure adequate signal transitions, bit stuffing is employed by the transmitting device when sending a packet on the USB. A 0 is inserted after every six consecutive 1s in the data stream before the data is NRZI encoded to force a transition in the NRZI data stream. This gives the receiver logic a data transition at least once every seven bit times to guarantee the data and clock lock. A waveform of the data encoding is shown in Figure 25-2.

Figure 25-2. USB Data Encoding



Bit stuffing is enabled beginning with the Sync Pattern and throughout the entire transmission. The data “one” that ends the Sync Pattern is counted as the first one in a sequence. Bit stuffing is always enforced, without exception. If required by the bit stuffing rules, a zero bit will be inserted even if it is the last bit before the end-of-packet (EOP) signal.

25.8 Bus Protocol

25.8.1 Bit Ordering

Bits are sent out onto the bus least significant bit (LSb) first, followed by next LSB, the most significant bit (MSb) last.

25.8.2 SYNC Field

All packets begin with a synchronization (SYNC) field, which is a coded sequence that generates a maximum edge transition density. The SYNC field appears on the bus as IDLE followed by the binary string “KJKJKJKK,” in its NRZI encoding. It is used by the input circuitry to align incoming data with the local clock and is defined to be eight bits in length. SYNC serves only as a synchronization mechanism and is not shown in the following packet diagrams in Section 25.8.3. The last two bits in the SYNC field are a marker that is used to identify the first bit of the PID. All subsequent bits in the packet must be indexed from this point.

25.8.3 Packet Field Formats

All packets have distinct start and end of packet delimiters. Full details are given in the *USB Specification*.

25.8.3.1 Packet Identifier Field

A packet identifier (PID) immediately follows the SYNC field of every USB packet. A PID consists of a four bit packet type field followed by a four-bit check field . The PID indicates the type of packet and, by inference, the format of the packet and the type of error detection applied to the packet. The four-bit check field of the PID insures reliable decoding of the PID so that the remainder of the packet is interpreted correctly. The PID check field is generated by performing a ones complement of the packet type field.

Any PID received with a failed check field or which decodes to a non-defined value is assumed to be corrupted and it, as well as the remainder of the packet, is ignored by the receiver.

PIDs are divided into four coding groups: token, data, handshake, and special, with the first two transmitted PID bits (PID<1:0>) indicating which group. This accounts for the distribution of PID codes.

25.8.4 Address Fields

Function endpoints are addressed using two fields: the function address field and the endpoint field. Full details on this are given in the *USB Specification*.

25.8.4.1 Address Field

The function address (ADDR) field specifies the function, via its address, that is either the source or destination of a data packet, depending on the value of the token PID. A total of 128 addresses are specified as ADDR[6:0]. The ADDR field is specified for IN, SETUP, and OUT tokens.

25.8.4.2 Endpoint Field

An additional four-bit endpoint (ENDP) field, permits more flexible addressing of functions in which more than one sub-channel is required. Endpoint numbers are function specific. The endpoint field is defined for IN, SETUP, and OUT token PIDs only.

25.8.5 Frame Number Field

The frame number field is an 11-bit field that increments by the host on a per frame basis. The frame number field rolls over upon reaching its maximum value of x7FF, and is sent only for SOF tokens at the start of each frame.

25.8.6 Data Field

The data field may range from 0 to 1023 bytes and must be an integral numbers of bytes. Table 25-31 lists the format for multiple bytes. Data bits within each byte are shifted out LSB first.

Table 25-31. Data Field

Bit	Data Sent
x - 1	Byte N - 1, D7
x	Byte N, D0
x + 1	Byte N, D1
x + 2	Byte N, D2
x + 3	Byte N, D3
x + 4	Byte N, D4

Table 25-31. Data Field

x + 5	Byte N, D5
x + 6	Byte N, D6
x + 7	Byte N, D7
x + 8	Byte N+1, D 0

25.8.7 Cyclic Redundancy Check (CRC)

CRC is used to protect the all non-PID fields in token and data packets. In this context, these fields are considered to be protected fields. Full details on this are given in the *USB Specification*.

25.9 Packet Formats

The USB protocol calls out several packet types: token, data, and handshake packets. Full details on this are given in the *USB Specification*.

25.10 USB Interrupts

25.10.1 Overview

There are two general groups of USB interrupt sources, those resulting from execution of transactions in the schedule, and those resulting from a CMI operation error. All transaction-based sources can be masked by software through the Interrupt Enable register. Additionally, individual transfer descriptors can be marked to generate an interrupt on completion.

When the CMI drives an interrupt for USB, it internally drives the PIRQA# pin for USB function #0(see Chapter 26.0, “USB 2.0 Host Controller: Bus 0, Device 29, Function 7”) until all sources of the interrupt are cleared. In order to accommodate some operating systems, the Interrupt Pin register must contain a different value for each function of this multi-function device.

25.10.2 Transaction-Based Interrupts

These interrupts are not signaled until after the status for the last complete transaction in the frame has been written back to host memory. This guarantees that software can safely process through (Frame List Current Index -1) when it is servicing an interrupt.

25.10.2.1 CRC Error/Time-out

A CRC/Time-out error occurs when a packet transmitted from the CMI to a USB device or a packet transmitted from a USB device to the CMI generates a CRC error. The CMI is informed of this event by a time out from the USB device or by the CMI’s CRC checker generating an error on reception of the packet. Additionally, a USB bus time-out occurs when USB devices do not respond to a transaction phase within 19 bit times of an EOP. Either of these conditions will cause the C_ERR field of the TD to be decremented.

When the C_ERR field decrements to zero, the following occurs:

- The Active bit in the TD is cleared
- The Stalled bit in the TD is set
- The CRC/Time-out bit in the TD is set.

- At the end of the frame, the USB Error Interrupt bit is set in the HC status register.

If the CRC/Time out interrupt is enabled in the Interrupt Enable register, a hardware interrupt will be signaled to the system.

25.10.2.2 Interrupt on Completion

Transfer Descriptors contain a bit that can be set to cause an interrupt on their completion. The completion of the transaction associated with that block causes the USB Interrupt bit in the HC Status Register to be set at the end of the frame in which the transfer completed. When a TD is encountered with the IOC bit set to 1, the IOC bit in the HC Status register is set to 1 at the end of the frame if the active bit in the TD is set to 0 (even if it was set to 0 when initially read).

If the IOC Enable bit of Interrupt Enable register (bit 2 of I/O offset 04h) is set, a hardware interrupt is signaled to the system. This status bit is set whether the TD completes successfully, or because of errors. If the completion is because of errors, the USB Error bit in the HC status register is also set.

25.10.2.3 Short Packet Detect

A transfer set is a collection of data which requires more than 1 USB transaction to completely move the data across the USB. An example might be a large print file which requires numerous TDs in multiple frames to completely transfer the data. Reception of a data packet that is less than the endpoint's Maximum Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer set, even if there are active TDs remaining for this transfer set. Setting the SPD bit in a TD indicates to the HC to set the USB Interrupt bit in the HC status register at the end of the frame in which this event occurs. This feature streamlines the processing of input on these transfer types. If the Short Packet Interrupt Enable bit in the Interrupt Enable register is set, a hardware interrupt is signaled to the system at the end of the frame where the event occurred.

25.10.2.4 Serial Bus Babble

When a device transmits on the USB for a time greater than its assigned Maximum Length, it is said to be babbling. This error results in the Active bit in the TD being cleared to 0 and the Stalled and Babble bits being set to one. The C_ERR field is not decremented for a babble. The USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame. A hardware interrupt is signaled to the system.

If an EOF babble was caused by the CMI (due to incorrect schedule for instance), the CMI will force a bit stuff error followed by an EOP and the start of the next frame.

25.10.2.5 Stalled

This event indicates that a device/endpoint returned a STALL handshake during a transaction or that the transaction ended in an error condition. The TDs Stalled bit is set and the Active bit is cleared. Reception of a STALL does not decrement the error counter. A hardware interrupt is signaled to the system.

25.10.2.6 Data Buffer Error

This event indicates that an overrun of incoming data or a under-run of outgoing data has occurred for this transaction. This would generally be caused by the CMI not being able to access required data buffers in memory within necessary latency requirements. Either of these conditions will cause the C_ERR field of the TD to be decremented.

When C_ERR decrements to zero, the Active bit in the TD is cleared, the Stalled bit is set, the USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame and a hardware interrupt is signaled to the system.

25.10.2.7 Bit Stuff Error

A bit stuff error results from the detection of a sequence of more that six 1s in a row within the incoming data stream. This will cause the C_ERR field of the TD to be decremented. When the C_ERR field decrements to 0, the Active bit in the TD is cleared to 0, the Stalled bit is set to 1, the USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame and a hardware interrupt is signaled to the system.

25.10.3 Non-Transaction Based Interrupts

If a CMI process error or system error occur, the CMI halts and immediately issues a hardware interrupt to the system.

25.10.3.1 Resume Received

This event indicates that the CMI received a RESUME signal from a device on the USB bus during a global suspend. If this interrupt is enabled in the Interrupt Enable register, a hardware interrupt will be signaled to the system allowing the USB to be brought out of the suspend state and returned to normal operation.

25.10.3.2 Process Error

The HC monitors certain critical fields during operation to ensure that it does not process corrupted data structures. These include checking for a valid PID and verifying that the MaxLength field is less than 1280. If it detects a condition that would indicate that it is processing corrupted data structures, it immediately halts processing, sets the HC Process Error bit in the HC Status register and signals a hardware interrupt to the system.

This interrupt cannot be disabled through the Interrupt Enable register.

25.10.3.3 Host System Error

The CMI sets this bit to 1 when a Parity Error, Master Abort, or Target Abort occurs on memory accesses. When this error occurs, the CMI clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. This interrupt cannot be disabled through the Interrupt Enable register.

25.10.3.4 Implementation Notes

1. If a bad PID is found, but the packet will not be run because there is not enough time left in the frame due to the size of the transfer or the packet time out, an error will not be flagged.
2. “If a bad PID is found, but the packet will not be run because the TD’s active bit is off, the host controller will not halt, and an error will not be flagged.”

25.11 USB Power Management

The Host Controller can be put into a suspended state and its power can be removed. This requires that certain bits of information are retained in the resume power plane of the CMI so that a device on a port may wake the system. Such a device may be a fax-modem, which will wake up the machine to receive a fax or take a voice message.

The following bits in I/O space are to be maintained when the CMI enters a low power state:

Table 25-32. Bits Maintained in Low Power States

Register	Offset	Bit	Description
Command	00h	3	Enter Global Suspend Mode (EGSM)
Status	02h	2	Resume Detect
Port Status and Control	10h and 12h	2	Port Enabled/Disabled
		6	Resume Detect
		8	Low-speed Device Attached
		12	Suspend

When the CMI detects a resume event on any of its ports, it sets the corresponding USB_STS bit in ACPI space. If USB is enabled as a wake/break event, the system will wake up and an SCI will be generated.

25.12 USB Legacy Keyboard Operation

When a USB keyboard is plugged into the system, and a standard keyboard is not, the system may not boot, and DOS legacy software will not run, because the keyboard will not be identified. The CMI implements a series of trapping operations which will snoop accesses that go to the keyboard controller, and put the expected data from the USB keyboard into the keyboard controller.

The following table summarizes the implementation of the bits in the USB Legacy Keyboard/Mouse Control Registers.

Table 25-33. USB Legacy Keyboard / Mouse Control Register Bit Implementation (Sheet 1 of 2)

Bit #	Bit Name	Summary	Details
15	SMI Caused by End of Pass-Through	Logically 1 bit for all controllers	This bit in all host controllers will be set at the same time and cleared at the same time. It is cleared whenever software writes a 1 to this bit in any of the classic USB host controllers. This bit may either be implemented separately for each controller or shared and aliased.
13	PCI Interrupt Enable	Independent enable	Each bit provides individual host control
12	SMI Caused by USB Interrupt	Independent status	Individual status bits for each controller
11	SMI Caused by Port 64 Write	Logically 1 bit for all controllers	This bit in all host controllers will be set at the same time and cleared at the same time. It is cleared whenever software writes a 1 to this bit in any of the classic USB host controllers. This bit may either be implemented separately for each controller or shared and aliased.
10	SMI Caused by Port 64 Read	Logically 1 bit for all controllers	This bit in all host controllers will be set at the same time and cleared at the same time. It is cleared whenever software writes a 1 to this bit in any of the classic USB host controllers. This bit may either be implemented separately for each controller or shared and aliased.
9	SMI Caused by Port 60 Write	Logically 1 bit for all controllers	This bit in all host controllers will be set at the same time and cleared at the same time. It is cleared whenever software writes a 1 to this bit in any of the classic USB host controllers. This bit may either be implemented separately for each controller or shared and aliased.

Table 25-33. USB Legacy Keyboard/Mouse Control Register Bit Implementation (Sheet 2 of 2)

Bit #	Bit Name	Summary	Details
8	SMI Caused by Port 60 Read	Logically 1 bit for all controllers	This bit in all host controllers will be set at the same time and cleared at the same time. It is cleared whenever software writes a 1 to this bit in any of the classic USB host controllers. This bit may either be implemented separately for each controller or shared and aliased.
7	SMI at End of Pass-Through Enable	Separate enables ORed together	This bit enables the generation of the SMI based on bit 15 within the same function. If bit 15 is implemented as a shared/aliased bit across all functions, then the bit 7s from all classic USB controllers are ORed together and used to enable the SMI based on bit 15.
6	Pass Through State	Logically 1 bit for all controllers	This bit in all host controllers reflects the state of the Pass-Through state machine. Software can force this bit to 0 by clearing the A20Gate Pass-Through Enable (bit 5) in *all* of the host controllers.
5	A20Gate Pass-Through Enable	ORed together to enable the pass-thru state machine	If any of these bits in the classic USB host controllers is set, then the IICH will enable the Legacy Keyboard A20Gate Pass-through sequence. This prevents the SMI status bits (11:8) from asserting in all controllers when the specific sequence of I/O cycles is observed.
4	SMI on USB IRQ	Independent Enable	Each bit provides individual host control
3	SMI on Port 64 Writes Enable	Separate enables ORed together	Each bit enables SMI generation if the corresponding bit 11 is set. If bit 11 is implemented as a shared/aliased bit across all functions, then the bit 3s from all classic USB controllers are ORed together and used to enable the SMI based on bit 11.
2	SMI on Port 64 Reads Enable	Separate enables ORed together	Each bit enables SMI generation if the corresponding bit 10 is set. If bit 10 is implemented as a shared/aliased bit across all functions, then the bit 2s from all classic USB controllers are ORed together and used to enable the SMI based on bit 10.
1	SMI on Port 60 Writes Enable	Separate enables ORed together	Each bit enables SMI generation if the corresponding bit 9 is set. If bit 9 is implemented as a shared/aliased bit across all functions, then the bit 1s from all classic USB controllers are ORed together and used to enable the SMI based on bit 9.
0	SMI on Port 60 Reads Enable	Separate enables ORed together	Each bit enables SMI generation if the corresponding bit 8 is set. If bit 8 is implemented as a shared/aliased bit across all functions, then the bit 0s from all classic USB controllers are ORed together and used to enable the SMI based on bit 8.

Note: The scheme described below assumes that the keyboard controller (8042 or equivalent) is on the LPC bus.

This legacy operation is performed through SMM space. The latched SMI source (60R, 60W, 64R, 64W) is available in the Status Register. Because the enable is after the latch, it is possible to check for other events that didn't necessarily cause an SMI. It is the software's responsibility to logically AND the value with the appropriate enable bits.

Note: SMI is generated before the LPC cycle completes on the IMCH/IICH interface to ensure that the processor doesn't complete the cycle before the SMI is observed. The logic will also need to block the accesses to the 8042.

If there is an external 8042, then this is accomplished by not activating the 8042 CS. This is done by logically ANDing the four enables (60R, 60W, 64R, 64W) with the 4 types of accesses to determine if 8042 CS should go active. An additional term is required for the “pass-through” case.

§ §

26.0 USB 2.0 Host Controller: Bus 0, Device 29, Function 7

26.1 Overview

This section focuses on CMI-specific implementation details of the *Universal Serial Bus (USB) Specification, Revision 2.0* and *Enhanced Host Controller Interface (EHCI) Specification* for Universal Serial Bus specifications.

Register address locations not shown in Section 26.2 must be treated as Reserved.

Note: All configuration registers in this section are in the Core Well and reset by a Core Well reset and the D3-to-D0 warm reset, except as noted.

CMI contains an Enhanced Host Controller Interface (EHCI) compliant host controller which supports up to two *USB Rev. 2.0 Specification*-compliant root ports. USB 2.0 allows data transfers up to 480 Mbits/s using the same pins as the two USB1 ports. CMI contains port-routing logic that determines whether a USB port is controlled by one of the UHCI controllers or by the EHCI controller. A USB 2.0-based Debug Port is also implemented in CMI. A summary of the key architectural differences between the USB 1.1 UHCI host controllers and the USB 2.0 EHCI host controller is shown in the table below:

Table 26-1. USB 1.1 and USB 2.0 Comparison

	USB 1.1 UHCI	USB 2.0 EHCI
Accessible by	I/O space	Memory Space
Memory Data Structure	Single linked list	Separated in to Periodic and Asynchronous lists
Differential Signaling Voltage	3.3 V	400 mV
Controllers	1	1
Ports per Controller	2	2

26.2 USB 2.0 PCI Configuration Registers

The default values are defined with an h for hex, a b for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved registers locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 26-2. Bus 0, Device 29, Function 7: Summary of USB (2.0) Controller PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID - Vendor ID Register" on page 979	8086h
02h	03h	"Offset 02h: DID - Device Identification Register" on page 979	5035h
04h	05h	"Offset 04h: CMD - Command Register" on page 980	0000h
06h	07h	"Offset 06h: DSR - Device Status Register" on page 981	0290h
08h	08h	"Offset 08h: RID - Revision ID Register" on page 983	Variable
09h	09h	"Offset 09h: PI - Programming Interface Register" on page 983	20h
0Ah	0Ah	"Offset 0Ah: SCC - Sub Class Code Register" on page 983	03h
0Bh	0Bh	"Offset 0Bh: BCC - Base Class Code Register" on page 984	0Ch
0Dh	0Dh	"Offset 0Dh: MLT - Master Latency Timer Register" on page 984	00h
10h	13h	"Offset 10h: MBAR - Memory Base Address Register" on page 985	00000000h
2Ch	2Dh	"Offset 2Ch: SSVID - USB 2.0 Subsystem Vendor ID Register" on page 985	XXXXh
2Eh	2Fh	"Offset 2Eh: SSID - USB 2.0 Subsystem ID Register" on page 986	XXXXh
34h	34h	"Offset 34h: CAP_PTR - Capabilities Pointer Register" on page 986	50h
3Ch	3Ch	"Offset 3Ch: ILINE - Interrupt Line Register" on page 987	00h
3Dh	3Dh	"Offset 3Dh: IPIN - Interrupt Pin Register" on page 987	Variable
50h	50h	"Offset 50h: PM_CID - PCI Power Management Capability ID Register" on page 987	01h
51h	51h	"Offset 51h: PM_NEXT - Next Item Pointer #1 Register" on page 988	58h
52h	53h	"Offset 52h: PM_CAP - Power Management Capabilities Register" on page 989	C9C2h
54h	55h	"Offset 54h: PM_CS - Power Management Control/Status Register" on page 990	0000h
58h	58h	"Offset 58h: DP_CID - Debug Port Capability ID Register" on page 991	0Ah
59h	59h	"Offset 59h: DP_NEXT - Next Item Pointer #2 Register" on page 991	00h
5Ah	5Bh	"Offset 5Ah: DP_BASE - Debug Port Base Offset Register" on page 991	20A0h
60h	60h	"Offset 60h: SBRN - Serial Bus Release Number Register" on page 992	20h
61h	61h	"Offset 61h: FLA - Frame Length Adjustment Register" on page 992	20h
62h	63h	"Offset 62h: PWC - Port Wake Capability Register" on page 993	01FFh
64h	65h	"Offset 64h: CUO - Classic USB Override Register" on page 994	0000h
68h	6Bh	"Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register" on page 994	00000001h
6Ch	6Fh	"Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register" on page 995	00000000h

Table 26-2. Bus 0, Device 29, Function 7: Summary of USB (2.0) Controller PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
70h	73h	"Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register" on page 997	00000000h
80h	80h	"Offset 80h: AC - Access Control Register" on page 999	00h
F8h	FBh	"Offset F8h: MANID - Manufacturer ID Register" on page 1000	00010F90h

26.2.1 Register Details

26.2.1.1 Offset 00h: VID - Vendor ID Register

Table 26-3. Offset 00h: VID - Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	VID	**Vendor ID:** This is a 16-bit value assigned to Intel.		8086h	RO

26.2.1.2 Offset 02h: DID - Device Identification Register

Table 26-4. Offset 02h: DID - Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5035h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	DID	**Device ID:** This is a 16-bit value assigned to the USB2 host controller.		5035h	RO

26.2.1.3 Offset 04h: CMD - Command Register

Table 26-5. Offset 04h: CMD - Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :11	Reserved	Reserved		00h	RO
10	INT_DIS	**Interrupt Disable:** 0 = The function is capable of generating interrupts. 1 = The function can not generate its interrupt to the interrupt controller. The corresponding Interrupt Status bit is not affected by the interrupt enable. This bit defaults to '0'. This bit is added as part of the *Conventional PCI 2.3 Specification*.		0h	RW
09	FBE	**Fast Back to Back Enable:** Reserved as '0'.		0h	RO
08	SERR_N_EN	SERR# Enable: 0 = The EHC is disabled from generating (internally) SERR# 1 = The EHC is capable of generating (internally) SERR# in the following cases: • Reception of status other than “Successful” on a memory read completion (if SERR on Aborts Enable is also set) • Detection of an address or command parity error and the Parity Error Response bit is set • Detection of a data parity error (when the data is going to the EHC) and the Parity Error Response bit is set • Since USB 2.0 logic does not support parity checking, bit 6 is never set.		0h	RW
07	WCC	**Wait Cycle Control:** Reserved as '0'.		0h	
06	PER	**Parity Error Response:** Reserved as ‘0’.		0h	
05	VPS	**VGA Palette Snoop:** Reserved as '0'.		0h	
04	PMWE	**Postable Memory Write Enable:** Reserved as '0'.		0h	
03	SCE	**Special Cycle Enable:** Reserved as '0'.		0h	
02	BME	**Bus Master Enable:** 0 = Clearing the BME bit shuts down the EHC DMA engines in the same manner that clearing the Run/ Stop does. However, the schedule status bits and the HCHalted bit do not change based on the BME value 1 = Acts as a master on the PCI bus for USB transfers. ***Notes:*** Notes on the EHC implementation: • Writes to change this bit occur immediately. Specifically, a write followed by a read will return the updated value. • When the BME bit is changed from 1 to 0, the EHC will cease accessing main memory within 2 microframes (250 µs). During this time, any number of reads and/ or writes to memory may occur.		0h	RW

Table 26-5. Offset 04h: CMD - Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	MSE	**Memory Space Enable:** This bit controls access to the USB 2.0 Memory Space registers. 0 = Accesses to the USB 2.0 registers are disabled 1 = Accesses to the USB 2.0 registers are enabled. The Base Address register for USB 2.0 must be programmed before this bit is set.		0h	RW
00	IOSE	**I/O Space Enable:** Reserved as '0'.		0h	

26.2.1.4 Offset 06h: DSR - Device Status Register

Note: For the writable bits, software must write a one to clear bits that are set. Writing a zero to the bit has no effect.

Table 26-6. Offset 06h: DSR - Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0290h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** 0 = No SERR# detected. 1 = Set when a parity error is detected on the internal interface to the USB host controller, regardless of the setting of bit 6 or bit 8 in the Command register or any other conditions. Software clears this bit by writing a '1' to this bit location.		0h	RWC
14	SSE	**Signaled System Error:** 0 = No SERR# detected. 1 = This bit is set whenever it signals SERR# (internally). The SERR_EN bit (bit 8 in the Command Register) must be 1 for this bit to be set. The following conditions can cause the generation of SERR#: • A parity error is seen on address, command, or data (if the data was targeting the EHC) on the internal interface to the USBe host controller due to a parity error on the system interface and bit 6 of the Command register is set to 1. • An EHC-initiated memory read results in a completion packet with a status other than successful on the system interface (if SERR on Aborts Enable is also set to 1). Software clears this bit by writing a '1' to this bit location.		0h	RWC

Table 26-6. Offset 06h: DSR - Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0290h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13	RMA	**Received Master-Abort Status:** 0 = No master abort received by EHC on a memory access. 1 = This bit is set when USB 2.0, as a master, receives a master-abort status on a memory access. This is treated as a Host Error and halts the DMA engines. This event can optionally generate an SERR# by setting the SERR# Enable bit and the SERR on Aborts Enable (bit 3, offset 84h). Software clears this bit by writing a '1' to this bit location.		0h	RWC
12	RTA	**Received Target Abort Status:** 0 = No target abort received by EHC on memory access. 1 = This bit is set when USB 2.0, as a master, receives a target abort status on a memory access. This is treated as a Host Error and halts the DMA engines. This event can optionally generate an SERR# by setting the SERR# Enable bit and the SERR on Aborts Enable (bit 3, offset 84h). Software clears this bit by writing a '1' to this bit location.		0h	RWC
11	STA	**Signaled Target-Abort Status:** This bit is used to indicate when the USB 2.0 function responds to a cycle with a target abort. This should never occur, so this bit is hard-wired to '0'.		0h	RO
10 :09	DEVT	**DEVSEL# Timing Status:** This 2-bit field defines the timing for DEVSEL# assertion.		01h	RO
08	DPD	**Master Data Parity Error Detected:** 0 = No data parity error detected on USB 2.0 read completion packet. 1 = This bit is set whenever a data parity error is detected on a USBe read completion packet on the internal interface to the USBe host controller and bit 6 of the Command register is set to 1. Software clears this bit by writing a '1' to this bit location.		0h	RWC
07	FB2BC	**Fast Back-to-Back Capable:** Reserved as '1'.		1	
06	Reserved	**User Definable Features:** Reserved as '0'.		0h	
05	C66	**66 MHz Capable:** Reserved as '0'.		0h	
04	CLIST	**Capabilities List:** Hardwired to '1' indicating that offset 34h contains a valid capabilities pointer.		1	RO
03	IS	**Interrupt Status:** This read-only bit reflects the state of this function's interrupt at the input of the enable/disable logic. 0 = This bit will be 0 when the interrupt is deasserted. 1 = This bit is a 1 when the interrupt is asserted. The value reported in this bit is independent of the value in the Interrupt Enable bit. This bit is added as part of the *PCI 2.3 Specification*.		0h	RO
02 :00	Reserved	Reserved.		000h	

26.2.1.5 Offset 08h: RID - Revision ID Register

Table 26-7. Offset 08h: RID - Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	RID	The value reported in this register depends on the value written to the Revision ID in Device 31, Function 0. This register follows the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

26.2.1.6 Offset 09h: PI - Programming Interface Register

Table 26-8. Offset 09h: PI - Programming Interface Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 09h **Offset End:** 09h	
Size: 8 bit	**Default:** 20h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	PIC	Program Interface Conforms: A value of 20h indicates that this USB 2.0 Host Controller conforms to the *EHCI Specification*.		20h	RO

26.2.1.7 Offset 0Ah: SCC - Sub Class Code Register

Table 26-9. Offset 0Ah: SCC - Sub Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:** 03h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	SCC	A value of 03h indicates that this is a Universal Serial Bus Host Controller.		03h	RO

26.2.1.8 Offset 0Bh: BCC - Base Class Code Register

Table 26-10. Offset 0Bh: BCC - Base Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:** 0Ch			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	BCC	A value of 0Ch indicates that this is a Serial Bus controller.		0Ch	RO

26.2.1.9 Offset 0Dh: MLT - Master Latency Timer Register

Table 26-11. Offset 0Dh: MLT - Master Latency Timer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	MLT	Because the USB 2.0 controller is internally implemented with arbitration on an internal interface, it does not need a master latency timer. The bits will be fixed at 0.		0h	RO

26.2.1.10 Offset 10h: MBAR - Memory Base Address Register

Table 26-12. Offset 10h: MBAR - Memory Base Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :10	BA	**Base Address:** Bits [31:10] correspond to memory address signals [31:10], respectively. This gives 1 KByte of relocatable memory space aligned to 1 KByte boundaries.		0h	RW
09 :04	Reserved	**Reserved**		0h	
03	PREF	**Prefetchable:** This bit is hardwired to 0 indicating that this range must not be prefetched.		0h	RO
02 :01	TPE	**Type:** This field is hardwired to 0 indicating that this range can be mapped anywhere within 32-bit address space.		00h	RO
00	RTE	**Resource Type Indicator:** This bit is hardwired to 0 indicating that the base address field in this register maps to memory space.		0h	RO

26.2.1.11 Offset 2Ch: SSVID - USB 2.0 Subsystem Vendor ID Register

Table 26-13. Offset 2Ch: SSVID - USB 2.0 Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** XXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SSVID	This register, in combination with the USB 2.0 Subsystem ID register, enables the operating system to distinguish each subsystem from the others. Writes to this register are enabled when the WRT_RDONLY bit (offset 80h, bit 0) is set to 1.		XXXXh	RW

26.2.1.12 Offset 2Eh: SSID - USB 2.0 Subsystem ID Register

Table 26-14. Offset 2Eh: SSID - USB 2.0 Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** XXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	SSID	BIOS sets the value in this register to identify the Subsystem ID. This register, in combination with the Subsystem Vendor ID register, enables the operating system to distinguish each subsystem from other(s). Writes to this register are enabled when the WRT_RDONLY bit (offset 80h, bit 0) is set to 1. Writes must be done as a single 16-bit cycle.		XXXXh	RW

26.2.1.13 Offset 34h: CAP_PTR - Capabilities Pointer Register

Table 26-15. Offset 34h: CAP_PTR - Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** 50h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	CAP_PTR	This register points to the starting offset of the USB 2.0 capabilities ranges.		50h	RO

26.2.1.14 Offset 3Ch: ILINE - Interrupt Line Register

Table 26-16. Offset 3Ch: ILINE - Interrupt Line Register

Description: Lockable: D3-to-D0					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	ILINE	**Interrupt line:** This data is not used. It is used as a scratchpad register to communicate to software the interrupt line that the interrupt pin is connected to.		00h	RW

26.2.1.15 Offset 3Dh: IPIN - Interrupt Pin Register

Table 26-17. Offset 3Dh: IPIN - Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 3Dh **Offset End:3D** 3Dh	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	IPIN	**Interrupt pin:** Bits 03:00 reflect the value of D29IP.EIP in configuration space. Bits 07:04 are hardwired to 0000b.		Variable	RO

26.2.1.16 Offset 50h: PM_CID - PCI Power Management Capability ID Register

Table 26-18. Offset 50h: PM_CID - PCI Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 50h **Offset End:** 50h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	PM_CID	A value of 01h indicates that this is a PCI Power Management capabilities field.		01h	RO

26.2.1.17 Offset 51h: PM_NEXT - Next Item Pointer #1 Register

Table 26-19. Offset 51h: PM_NEXT - Next Item Pointer #1 Register

Description: Lockable:Not D3-to-DO					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 51h **Offset End:** 51h	
Size: 8 bit	**Default:** 58h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	PM_NEXT	This register defaults to 58h, which indicates that the next capability registers begin at configuration offset 58h. This register is writable when the WRT_RDONLY bit is set. This allows BIOS to effectively hide the Debug Port capability registers, if necessary. This register must only be written during system initialization before the plug-and-play software has enabled any master-initiated traffic. Only values of 58h (Debug Port visible) and 00h (Debug Port invisible) are expected to be programmed in this register.		58h	RW

26.2.1.18 Offset 52h: PM_CAP - Power Management Capabilities Register

Normally, this register is read-only to report capabilities to the power management software. In order to report different power management capabilities depending on the system in which CMI is used, bits 15:11 and 08:06 in this register are writable when the WRT_RDONLY bit in Table 26-32, “Offset 80h: AC - Access Control Register” on page 999 is set. The value written to this register does not affect the hardware other than changing the value returned during a read.

Reset: core well, but not D3-to-D0.

Table 26-20. Offset 52h: PM_CAP - Power Management Capabilities Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 52h **Offset End:** 53h	
Size: 16 bit	**Default:** C9C2h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :11	PME_Support	This 5-bit field indicates the power states in which the function may assert PME#. The CMI EHC does not support the D1 or D2 states. For all other states, the CMI EHC is capable of generating PME#. Software must never need to modify this field. Note: These bits are writable (RW) when D29:F7:80h, bit 0 is 1		11001b	RWL
10	D2_Support	The D2 state is not supported. It is hardwired to ‘0’.		0b	RO
09	D1_Support	The D1 state is not supported.It is hardwired to ‘0’.		0b	RO
08 :06	Aux_Current	EHC reports 375 mA maximum Suspend well current required when in the $D3_{COLD}$ state. This value can be written by BIOS when a more accurate value is known. Note: These bits are writable (RW) when D29:F7:80h, bit 0 is 1		111b	RWL
05	DSI	Reports 0, indicating that no device-specific initialization is required. It is hardwired to ‘0’.		0b	RO
04	Reserved	Reserved.		0b	RO
03	PME_Clock	Reports 0, indicating that no PCI clock is required to generate PME#. It is hardwired to ‘0’.		0b	RO
02 :00	Version	Reports 010, indicating that it complies with Revision 1.1 of the *PCI Power Management Specification*. It is hardwired to ‘010’.		010b	RO

26.2.1.19 Offset 54h: PM_CS - Power Management Control/Status Register

Table 26-21. Offset 54h: PM_CS - Power Management Control/Status Register

Description: Reset (bits 15:08):Suspend Well, not D3-to-D0 nor core well reset. Reset (bits 1:0):Core Well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 54h **Offset End:** 55h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_Status	0 = Writing a 0 has no effect. 1 = Set when EHC would normally assert the PME# signal independent of the state of the PME_En bit. ***Note:*** Writing a 1 to this bit will clear it and cause the internal PME to deassert (if enabled). This bit must be explicitly cleared by the operating system each time the operating system is loaded.		0h	RWC
14 :13	Data_Scale	Hardwired to “00” because it does not support the associated Data register.		00h	RO
12 :09	Data_Select	Hardwired to “0000” because it does not support the associated Data register.		0000h	RO
08	PME_En	A ‘1’ enables EHC to generate an internal PME signal when PME_Status is ‘1’. This bit must be explicitly cleared by the operating system each time it is initially loaded.		0h	RW
07 :02	Reserved	Reserved		0h	
01 :00	PowerState	This 2-bit field is used both to determine the current power state of EHC function and to set a new power state. The definition of the field values are: 00b – D0 state 11b – $D3_{HOT}$ state If software attempts to write a value of 10b or 01b in to this field, the write operation must complete normally; however, the data is discarded and no state change occurs. When in the D3hot state, accesses to the EHC memory range must not be accessible, but the configuration space must still be accessible. When not in the D0 state, the generation of the interrupt output is blocked. Specifically, the PIRQ[A] is not asserted by CMI when not in the D0 state. When software changes this value from the $D3_{HOT}$ state to the D0 state, an internal warm (soft) reset is generated, and software must reinitialize the function.		00h	RW

26.2.1.20 Offset 58h: DP_CID - Debug Port Capability ID Register

Table 26-22. Offset 58h: DP_CID - Debug Port Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 58h **Offset End:** 58h	
Size: 8 bit	**Default:** 0Ah			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	DP_CID	This register is hardwired to 0Ah which indicates that this is the start of a Debug Port Capability structure.		0Ah	RO

26.2.1.21 Offset 59h: DP_NEXT - Next Item Pointer #2 Register

Table 26-23. Offset 59h: DP_NEXT - Next Item Pointer #2 Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 59h **Offset End:** 59h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	DP_NEXT	This register is hardwired to 00h, which indicates there are no more capability structures in this function.		0h	RO

26.2.1.22 Offset 5Ah: DP_BASE - Debug Port Base Offset Register

This register is hardwired to 20A0h, which indicates that the Debug Port Registers begin at offset A0h in the USB 2.0 function's memory space.

Table 26-24. Offset 5Ah: DP_BASE - Debug Port Base Offset Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 5Ah **Offset End:** 5Bh	
Size: 16 bit	**Default:** 20A0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :13	BNBR	**BAR Number:** This field is hardwired to 001b to indicate the memory BAR at offset 10h in the EHCI configuration space.		001h	RO
12 :00	DPO	**Debug Port Offset:** This field is hardwired to 0A0h to indicate that the debug port registers begin at offset A0h in the EHCI memory range.		0A0h	RO

26.2.1.23 Offset 60h: SBRN - Serial Bus Release Number Register

Table 26-25. Offset 60h: SBRN - Serial Bus Release Number Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 60h **Offset End:** 60h	
Size: 8 bit	**Default:** 20h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	SBRN	A value of 20h indicates that this controller follows USB 2.0.		20h	RO

26.2.1.24 Offset 61h: FLA - Frame Length Adjustment Register

This feature is used to adjust any offset from the clock source that generates the clock that drives the SOF counter. When a new value is written into these six bits, the length of the frame is adjusted. Its initial programmed value is system dependent based on the accuracy of hardware USB clock and is initialized by system BIOS. This register must only be modified when the *HChalted* bit in the USBSTS register is a one. Changing value of this register while the host controller is operating yields undefined results. It must not be reprogrammed by USB system software unless the default or BIOS programmed values are incorrect, or the system is restoring the register while returning from a suspended state.

Table 26-26. Offset 61h: FLA - Frame Length Adjustment Register

Description: Reset: Suspend well, not D3-to-D0 nor core well reset					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 61h **Offset End:** 61h	
Size: 8 bit	**Default:** 20h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :06	Reserved	Reserved.		00h	
05 :00	FLTV	**Frame Length Timing Value:** Each decimal value change to this register corresponds to 16 high-speed bit times. The SOF cycle time (number of SOF counter clock periods to generate a SOF microframe length) is equal to 59488 + value in this field. The default value is decimal 32 (20h), which gives a SOF cycle time of 60000. **Frame Length** (# High Speed bit times) FLADJ Value (decimal) (decimal) 59488 0 (00h) 59504 1 (01h) 59520 2 (02h) ... 59984 31 (1Fh) 60000 32 (20h) ... 60480 62(3Eh) 60496 63 (3Fh)		100000b	RW

26.2.1.25 Offset 62h: PWC - Port Wake Capability Register

This register is in the suspend power well. The intended use of this register is to establish a policy about which ports are to be used for wake events. Bit positions 1-8 in the mask correspond to a physical port implemented on the current EHCI controller. A one in a bit position indicates that a device connected below the port can be enabled as a wake-up device and the port may be enabled for disconnect/connect or overcurrent events as wake-up events. This is an information-only mask register. The bits in this register DO NOT affect the actual operation of the EHCI host controller. The system-specific policy can be established by BIOS initializing this register to a system-specific value. System software uses the information in this register when enabling devices and ports for remote wake-up.

Note: There is no support for wake from USB when in S3/S4/S5.

Table 26-27. Offset 62h: PWC - Port Wake Capability Register

Description: Reset: suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 62h **Offset End:** 63h	
Size: 16 bit	**Default:** 01FFh			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :09	Reserved	Reserved.		0000000b	
08 03	Reserved	**Reserved**		111111b	RW
02 :01	PWU	**Port Wake Up Capability Mask:** Bit positions 1 through 2 correspond to a physical port implemented on this host controller. For example, bit position 1 corresponds to port 1, position 2 port 2, etc.		11b	RW
00	PWI	**Port Wake Implemented:** 0 = Indicates that this register is not supported by software. 1 = Indicates that this register is supported by software.		1b	RW

26.2.1.26 Offset 64h: CUO - Classic USB Override Register

This 16-bit register provides a bit corresponding to each of the ports on the EHCI host controller (The *EHCI Specification* supports up to 16 ports). When a bit is set to '1', the corresponding USB port is routed to the classic (UHCI) host controller and will only operate using the classic signaling rates. The feature is implemented with the following requirements:

- The associated Port Owner bit does *not* reflect the value in this Override register. This guarantees compatibility with EHCI drivers.
- BIOS must only write to this register during initialization (while the Configured Flag is '0').
- The register is implemented in the Suspend well to maintain port routing when the core power goes down.

- When a '1' is present in the Override register, then the classic controller operates the port regardless of the EHCI port routing logic. The corresponding EHCI port will always appear disconnected in this mode.

Note: EHCI test modes will not work on a port that has been overridden by this register.

- Port 0 must never be programmed to the Classic USB Override mode. This is because the Debug Port is used on Port 0 and the two modes conflict with each other.

Table 26-28. Offset 64h: CUO - Classic USB Override Register

Description: Reset:Suspend well, and not D3-to-D0, HCRESET nor core well.					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 64h **Offset End:** 65h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :08	Reserved	Reserved. These bits are reserved for future use; Reads return 0.		00h	
07 :02	Reserved	Reserved		000000b	RW
01 :1	CUSBPO	**Classic USB Port Owner:** A '1' in a bit position forces the corresponding USB port to the classic host controller.		00b	RW
00	Reserved	Reserved		0h	

26.2.1.27 Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register

Table 26-29. Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register (Sheet 1 of 2)

Description: Reset: suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :25	Reserved	Reserved. Hardwired to 00h.		00h	
24	HC_OS	**Host Controller OS Owned Semaphore:** System software sets this bit to request ownership of the EHCI controller. 0 = Ownership of the EHCI controller is not obtained. 1 = Ownership is obtained when this bit reads as 1 and the HC BIOS Owned Semaphore bit reads as clear.		0h	RW
23 :17	Reserved	Reserved.		00h	

Table 26-29. Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register (Sheet 2 of 2)

Description: Reset: suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
16	HC_BIOS	**Host Controller BIOS Owned Semaphore:** The BIOS sets this bit to establish ownership of the EHCI controller. System BIOS will clear this bit in response to a request for ownership of the EHCI controller by system software.		0h	RW
15 :08	NEHCI	**Next EHCI Capability Pointer:** A value of 00h indicates that there are no EHCI Extended Capability structures in this device.		00h	RO
07 :00	CAPID	**Capability ID:** A value of 01h indicates that this EHCI Extended Capability is the Legacy Support Capability.		01h	RO

26.2.1.28 Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register

Writing a '1' to that bit location clears bits that are marked as Read/Write-Clear (RWC).

Table 26-30. Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register (Sheet 1 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 6Ch **Offset End:** 6Fh	
Size: 32bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	SMI_BAR	**SMI on BAR:** 0 = Base Address Register (BAR) not written. 1 = This bit is set to 1 when the Base Address Register (BAR) is written.		0h	RWC
30	SMI_PCMD	**SMI on PCI Command:** This bit is set to '1' whenever the PCI Command Register is written. 0 = PCI Command (PCICMD) Register Not written. 1 = This bit is set to 1 when the PCI Command (PCICMD) Register is written.		0h	RWC
29	SMI_OSC	**SMI on OS Ownership Change:** 0 = No HC OS Owned Semaphore bit change. 1 = This bit is set to 1 when the HC OS Owned Semaphore bit in the LEG_EXT_CAP register (D29:F7:68h, bit 24) transitions from 1 to 0 or 0 to 1.		0h	RWC
28 :22	Reserved	Reserved. Hardwired to 0.		00h	
21	SMI_AA	**SMI on Async Advance:** Shadow bit of the *Interrupt on Async Advance* bit in the USB 2.0STS register. To clear this bit, system software must write a one to the *Interrupt on Async Advance* bit in the USB 2.0STS register.		0h	RO

Table 26-30. Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register (Sheet 2 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 6Ch **Offset End:** 6Fh	
Size: 32bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
20	SMI_HSE	**SMI on Host System Error:** Shadow bit of *Host System Error* bit in the USB 2.0STS. To clear this bit, system software must write a one to the *Host System Error* bit in the USB 2.0STS register.		0h	RO
19	SMI_FLR	**SMI on Frame List Rollover:** Shadow bit of *Frame List Rollover* bit in the USB 2.0STS register. To clear this bit, system software must write a one to the *Frame List Rollover* bit in the USB 2.0STS register.		0h	RO
18	SMI_PCD	**SMI on Port Change Detect:** Shadow bit of *Port Change Detect* bit in the USB 2.0STS register. To clear this bit, system software must write a one to the *Port Change Detect* bit in the USB 2.0STS register.		0h	RO
17	SMI_USBER	**SMI on USB Error:** Shadow bit of *USB Error Interrupt* (USBERRINT) bit in the USB 2.0STS register. To clear this bit, system software must write a one to the *USB Error Interrupt* bit in the USB 2.0STS register.		0h	RO
16	SMI_USBC	**SMI on USB Complete:** Shadow bit of *USB Interrupt* (USBINT) bit in the USB 2.0STS register. To clear this bit, system software must write a one to the *USB Interrupt* bit in the USB 2.0STS register.		0h	RO
15	SMI_BAREN	**SMI on BAR Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on BAR (D29:F7:6Ch, bit 31) is 1, then the host controller will issue an SMI.		0h	RW
14	SMI_PCIEN	**SMI on PCI Command Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on PCI Command (D29:F7:6Ch, bit 31) is 1, then the host controller will issue an SMI.		0h	RW
13	SMI_OSEN	**SMI on OS Ownership Enable:** 0 = Disable. 1 = Enable. When this bit is a 1 AND the OS Ownership Change bit (D29:F7:6Ch, bit 29) is 1, the host controller will issue an SMI.		0h	RW
12 :06	Reserved	Reserved. Hardwired to 0.		00h	
05	SMI_AAEN	**SMI on Async Advance Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on Async Advance bit (D29:F7:6Ch, bit 21) is a 1, the host controller will issue an SMI immediately.		0h	RW
04	SMI_HSEN	**SMI on Host System Error Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on Host System Error (D29:F7:6Ch, bit 20) is a 1, the host controller will issue an SMI.		0h	RW
03	SMI_FLREN	**SMI on Frame List Rollover Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on Frame List Rollover bit (D29:F7:6Ch, bit 19) is a 1, the host controller will issue an SMI.		0h	RW

Table 26-30. Offset 6Ch: ULSCS - USB 2.0 Legacy Support Control/Status Register (Sheet 3 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 6Ch **Offset End:** 6Fh	
Size: 32bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	SMI_PCEN	**SMI on Port Change Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on Port Change Detect bit (D29:F7:6Ch, bit 18) is a 1, the host controller will issue an SMI.		0h	RW
01	SMI_USBEN	**SMI on USB Error Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on USB Error bit (D29:F7:6Ch, bit 17) is a 1, the host controller will issue an SMI immediately.		0h	RW
00	SMI_USBCE	**SMI on USB Complete Enable:** 0 = Disable. 1 = Enable. When this bit is a 1, and the SMI on USB Complete bit (D29:F7:6Ch, bit 16) is a 1, the host controller will issue an SMI immediately.		0h	RW

26.2.1.29 Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register

This register provides a mechanism for BIOS to provide USB 2.0 related bug fixes and workarounds. Writing a '1' to that bit location clears bits that are marked as Read/Write/Clear (RW/C). Software must clear all SMI status bits prior to setting the global SMI enable bit and individual SMI enable bit to prevent spurious SMI when returning from a power down.

Table 26-31. Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register (Sheet 1 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :30	Reserved	Reserved. Hardwired to 0.		00b	
29 :26	Reserved	Reserved		0000b	RW
25 :24	Reserved	Reserved		00b	RWC
23 :22	SMI_PO	**SMI on PortOwner:** Bits 23:22 correspond to the Port Owner bits for ports 1 (22) through 02 (23). These bits are set to '1' whenever the associated Port Owner bits transition from 0->1 or 1->0. Software clears these bits by writing a one.		00b	RWC

Table 26-31. Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register (Sheet 2 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:29:7		**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
21	SMI_ PMCSR	**SMI on PMCSR:** 0 = Power State bits not modified. 1 = Software modified the Power State bits in the Power Management Control/Status (PMCSR) register (D29:F7:54h).		0b	RWC
20	SMI_ASYNC	**SMI on Async:** 0 = No Async Schedule Enable bit change 1 = Async Schedule Enable bit transitioned from 1 to 0 or 0 to 1.		0b	RWC
19	SMI_PER	**SMI on Periodic:** 0 = No Periodic Schedule Enable bit change. 1 = Periodic Schedule Enable bit transitions from 1 to 0 or 0 to 1.		0b	RWC
18	SMI_CF	**SMI on CF:** 0 = No Configure Flag (CF) change. 1 = Configure Flag (CF) transitions from 1 to 0 or 0 to 1.		0b	RWC
17	SMI_HCH	**SMI on HCHalted:** 0 = HCHalted did not transition to 1 (as a result of the Run/Stop bit being cleared). 1 = HCHalted transitions to 1 (as a result of the Run/Stop bit being cleared).		0b	RWC
16	SMI_HCR	**SMI on HCReset:** 0 = HCRESET did not transition to 1. 1 = HCRESET transitioned to 1.		0h	RWC
15 :14	Reserved	Reserved.		00b	
13 :08	Reserved	Reserved		000000b	RW
07 :06	SMI_POEN	**SMI on PortOwner Enable:** When any of these bits are '1' and the corresponding SMI on PortOwner bits are '1', then the host controller will issue an SMI. Unused ports must have their corresponding bits cleared.		00b	RW
05	SMI_PMSCREN	**SMI on PMSCR Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on PMSCR is 1, then the host controller will issue an SMI.		0b	RW
04	SMI_ASYEN	**SMI on Async Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on Async is 1, then the host controller will issue an SMI.		0b	RW
03	SMI_PEREN	**SMI on Periodic Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on Periodic is 1, then the host controller will issue an SMI.		0b	RW

Table 26-31. Offset 70h: ISU2SMI - Intel Specific USB 2.0 SMI Register (Sheet 3 of 3)

Description: Lockable: Suspend well, and not D3-to-D0 warm reset nor core well.					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 70h **Offset End:** 73h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	SMI_CFEN	**SMI on CF Enable:** 0 = Disable. 1 = Enable. When this bit is 1 and SMI on CF is 1, then the host controller will issue an SMI.		0b	RW
01	SMI_HCHEN	**SMI on HCHalted Enable:** 0 = Disable. 1 = Enable. When this bit is a 1 and SMI on HCHalted is 1, then the host controller will issue an SMI.		0b	RW
00	SMI_HCREN	**SMI on HCReset Enable:** 0 = Disable. 1 = Enable. When this bit is a 1 and SMI on HCReset is 1, then host controller will issue an SMI.		0b	RW

26.2.1.30 Offset 80h: AC - Access Control Register

Table 26-32. Offset 80h: AC - Access Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** 80h **Offset End:** 80h	
Size: 8 bit	**Default:** 00h			**Power Well:**	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :01	Reserved	Reserved		00h	
00	WRT_RDONLY	Write Read only: 0 = Disables a select group of normally read-only registers in the EHC function to be written by software. 1 = Enables a select group of normally read-only registers in the EHC function to be written by software. Registers that may only be written when this mode is entered are noted in the summary tables and detailed description as “Read/Write-Special”. The registers fall into two categories: a. System-configured parameters b. Status bits		0h	RW

26.2.1.31 Offset F8h: MANID - Manufacturer ID Register

Table 26-33. Offset F8h: MANID - Manufacturer ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:29:7	**Offset Start:** F8h **Offset End:** FBh	
Size: 32 bit	**Default:** 00010F90h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved		00h	
23 :16	SID	Stepping ID: This field increments for each stepping of the part. This field can be used by software to differentiate steppings when the Revision ID may not change. Implementation Note: A single Stepping ID can be implemented that is readable from all functions in the chip because all of them are incremented in lock-step ***Note:*** 00h for A0 stepping ***Note:*** 01h for B0 stepping See Device 31, Function 0, Offset F8h for the reported value.		01h	RO
15 :08	MID	Manufacturing ID: Indicates 0Fh = Intel		0Fh	RO
07 :00	Reserved	Reserved		90h	

26.3 USB 2.0 Memory-Mapped I/O Registers

The USB 2.0 EHCI memory-mapped I/O space is composed of two sets of registers: Capability Registers and Operational Registers. The base address of the address space that these registers materialize in is set by the MBAR BAR in the PCI configuration header (see Section 26.2.1.10, "Offset 10h: MBAR - Memory Base Address Register" on page 985).

The EHCI controller does not support as a target memory transactions that are locked transactions. Attempting to access the EHCI controller Memory-Mapped I/O space using locked memory transactions will result in undefined behavior.

Note: When the USB 2.0 function is in the D3 PCI power state, accesses to the USB 2.0 memory range are ignored and will result in a master abort. Similarly, if the Memory Space Enable (MSE) bit is not set in the Command register in configuration space, the memory range will not be decoded by the Enhanced Host Controller (EHC). If the MSE bit is not set, then CMI must default to allowing any memory accesses for the range specified in the BAR to go to LPC. This is because the range may not be valid and, therefore, the cycle must be made available to any other targets that may be currently using that range.

Table 26-34. Bus 0, Device 29, Function 7: Summary of USB (2.0) Controller Configuration Registers Mapped Through MBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: CAPLENGTH - Capability Length Register" on page 1002	20h
02h	03h	"Offset 02h: HCIVERSION - Host Controller Interface Version Number Register" on page 1003	0100h
04h	07h	"Offset 04h: HCSPARAMS - Host Controller Structural Parameters Register" on page 1003	01001202h
08h	0Bh	"Offset 08h: HCCPARAMS - Host Controller Capability Parameters Register" on page 1004	00006871h
20h	23h	"Offset 20h: USB2CMD - USB 2.0 Command Register" on page 1007	00080000h
24h	27h	"Offset 24h: USB2STS - USB 2.0 Status Register" on page 1009	00001000h
28h	2Bh	"Offset 28h: USB2INTR - USB 2.0 Interrupt Enable Register" on page 1012	00000000h
2Ch	2Fh	"Offset 2Ch: FRINDEX - Frame Index Register" on page 1013	00000000h
30h	33h	"Offset 30h: CTRLDSSEGMENT - Control Data Structure Segment Register" on page 1014	00000000h
34h	37h	"Offset 34h: PERIODICLISTBASE - Periodic Frame List Base Address Register" on page 1014	00000XXXh
38h	3Bh	"Offset 38h: ASYNCLISTADDR - Current Asynchronous List Address Register" on page 1015	00000000h
60h	63h	"Offset 60h: CONFIGFLAG - Configure Flag Register" on page 1015	00000000h
64h	67h	"Offset 64h: PORTSC - Port N Status and Control Register" on page 1016	00003000h
68h	6Bh	"Offset 64h: PORTSC - Port N Status and Control Register" on page 1016	00003000h
A0h	A3h	"Offset A0h: CNTL_STS - Control/Status Register" on page 1037	00000000h
A4h	A4h	"Offset A4h: USBPID - USB PIDs Register" on page 1039	00000000h
A8h	AFh	"Offset A8h: DATABUF - Data Buffer Bytes 7:0" on page 1039	0000000000000000h
B0h	B0h	"Offset B0h: CONFIG - Configuration Register" on page 1040	00007F01h

26.3.1 Host Controller Capability Register Details

These registers specify the limits, restrictions and capabilities of the host controller implementation.

Within the Host Controller Capability Registers, only the Structural Parameters register is writable. This register is implemented in the Suspend well and is only reset by the standard suspend-well hardware reset, not by HCRESET or the D3-to-D0 reset.

26.3.1.1 Offset 00h: CAPLENGTH - Capability Length Register

This register is used as an offset to add to the Memory Base Register to find the beginning of the Operational Register Space. This is fixed at 20h, indicating that the Operation Registers begin at offset 20h.

Table 26-35. Offset 00h: CAPLENGTH - Capability Length Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 20h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	CRLV	**Capability Register Length Value:** This register is used as an offset to add to the Memory Base Register (D29:F7:10h) to find the beginning of the Operational Register Space. This field is hardwired to 20h indicating that the Operation Registers begin at offset 20h.		20h	RO

26.3.1.2 Offset 02h: HCIVERSION - Host Controller Interface Version Number Register

This is a 2-byte register containing a BCD encoding of the version number of interface to which this host controller interface conforms.

Table 26-36. Offset 02h: HCIVERSION - Host Controller Interface Version Number Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 0100h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :00	HCIVERSION	**Host Controller Interface Version Number:** This is a two-byte register containing a BCD encoding of the version number of interface to which this host controller interface conforms.		0100h	RO

26.3.1.3 Offset 04h: HCSPARAMS - Host Controller Structural Parameters Register

This is a set of fields that are structural parameters: Number of downstream ports, etc. Some fields in this register are writable when the WRT_RDONLY bit is set. Fields that are described as “hardwired” are never writable. This register is implemented in the suspend well to avoid having to reload the parameters following a system sleep state in which the core power is removed.

Table 26-37. Offset 04h: HCSPARAMS - Host Controller Structural Parameters Register (Sheet 1 of 2)

Description: Reset: Suspend well reset, but not D3-to-D0 reset or HCRESET.					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 01001202h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved.		00h	
23 :20	DP_N	**Debug Port Number:** Hardwired to 1h, indicating that the Debug Port is on the lowest numbered port.		0001b	RO
19 :17	Reserved	Reserved.		000b	
16	P_INDICATOR	**Port Indicators:** This bit indicates whether the ports support port indicator control. CMI USB 2.0 Controller does not support Port Indicator LEDs, and this bit is hard wired to ‘0’.		0h	RO
15 :12	N_CC	**Number of Companion Controllers:** This field indicates the number of companion controllers associated with this USB 2.0 host controller. A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports. A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.		0001b	RWS

Table 26-37. Offset 04h: HCSPARAMS - Host Controller Structural Parameters Register (Sheet 2 of 2)

Description: Reset: Suspend well reset, but not D3-to-D0 reset or HCRESET.					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** 04h **Offset End:** 07h	
Size: 32 bit	**Default:** 01001202h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 :08	N_PCC	**Number of Ports per Companion Controller:** This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software. Hardwired to 2h.		0010b	RO
07 :04	Reserved	Reserved.		00h	
03 :00	N_PORTS	This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1H to 2H 2h is reported by default. However, software may write a value less than 2 for some platform configurations. A zero in this field is undefined.		0010b	RWS

26.3.1.4 Offset 08h: HCCPARAMS - Host Controller Capability Parameters Register

This register provides general mode information that affects the generation of the data structure in memory.

Table 26-38. Offset 08h: HCCPARAMS - Host Controller Capability Parameters Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 00006871h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		0000h	
15 :08	EECP	**EHCI Extended Capabilities Pointer:** This field is hardwired to 68h, indicating that the EHCI capabilities list exists and begins at offset 68h in the PCI configuration space.		68h	RO
07 :04	Reserved	Reserved		7h	RO
03	Reserved	Reserved		0h	

Table 26-38. Offset 08h: HCCPARAMS - Host Controller Capability Parameters Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 08h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 00006871h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	ASPC	**Asynchronous Schedule Park Capability:** This bit is hardwired to 0 indicating that the Host Controller does not support this optional feature.		0h	RO
01	PFLF	**Programmable Frame List Flag:** 0 = If this bit is set to a zero, then system software must use a frame list length of 1024 elements with this host controller. The USBCMD register *Frame List Size* field is a read-only register and must be set to zero. 1 = If set to a one, then system software can specify and use a smaller frame list and configure the host controller via the USBCMD register *Frame List Size* field. The frame list must always be aligned on a 4K page boundary. This requirement ensures that the frame list is always physically contiguous. Different frame list lengths are not supported. This bit is read-only '0'.		0h	RO
00	ADD_CAP	**64-bit Addressing Capability:** This field documents the addressing range capability of this implementation. The value of this field determines whether software must use the 32-bit or 64-bit data structures. Values for this field have the following interpretation: 0 = Data structures using 32-bit address memory pointers 1 = Data structures using 64-bit address memory pointers Only 64-bit addressing is supported. This bit is read-only '1'. Only 44 bits of addressing is supported. Bits 63:44 will always be 0 on cycles generated to memory.		1h	RO

26.3.2 Host Controller Operational Register Details

This section defines the enhanced host controller operational registers. These registers are located after the capabilities registers. The operational register base must be DWord aligned and is calculated by adding the value in the first capabilities register to the base address of the enhanced host controller register address space. In the following text, the offset is relative to the Memory Base Register. All registers are 32 bits in length. Software must read and write these registers using only Dword accesses.

These registers are divided into two sets. The first set at offsets 20h to 3Fh are implemented in the core power well. Unless otherwise noted, the core-well registers are reset by the assertion of any of the following:

- core well hardware reset
- HCRESET
- D3-to-D0 reset

The second set at offsets 60h to the end of the implemented register space are implemented in the Suspend power well. Unless otherwise noted, the core-well registers are reset by the assertion of either of the following:

- suspend well hardware reset
- HCRESET

The default values are defined with an h for hex, a b for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved registers locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 26-39. Host Controller Operational Register Details Summary Table

MEM_BASE + Offset		Symbol	Register Name/Function	Default	Special Notes	Access
Start	End					
20	23h	USB 2.0CMD	USB 2.0 Command Register	00080000h		RW
24	27h	USB 2.0STS	USB 2.0 Status Register	00001000h		RWC, RO
28	2Bh	USB 2.0INTR	USB 2.0 Interrupt Enable Register	00000000h		RW
2C	2Fh	FRINDEX	USB 2.0 Frame Index Register	00000000h		RW, RO
30	33h	CTRLDSSEGMENT	Control Data Structure Segment Register	00000000h		RW
34	37h	PERIODICLISTBASE	Period Frame List Base Address Register	00000000h		RW
38	3Bh	ASYNCLISTADDR	Next Asynchronous List Address Register	00000000h		RW, RO
60	63h	CONFIG FLAG	Configure Flag Register	00000000h	Suspend	RW, RO
64	67h	PORTSC	Port 1 Status and Control Register	00003000h	Suspend	RW
68	6Bh	PORTSC	Port 2 Status and Control Register	00003000h	Suspend	RW
A0	B3h		Debug Port Registers (see Section 26.13.2, “Debug Port Register Details”)			RW

26.3.2.1 Offset 20h: USB2CMD - USB 2.0 Command Register

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed.

Table 26-40. Offset 20h: USB2CMD - USB 2.0 Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** 20h **Offset End:** 23h	
Size: 32 bit	**Default:** 00080000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved.		00h	
23 :16	ITC	**Interrupt Threshold Control:** Default 08h. This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined. Value Maximum Interrupt Interval 00h Reserved 01h 1 microframe 02h 2 microframes 04h 4 microframes 08h 8 microframes (default, equates to 1 ms) 10h 16 microframes (2 ms) 20h 32 microframes (4 ms) 40h 64 microframes (8 ms) Refer to Section 4 in the *EHCI Specification* for interrupts affected by this field.		08h	RW
15 :12	Reserved	Reserved.		0h	
11 :08	UAPM	**Unimplemented Asynchronous Park Mode Bits:** This field is hardwired to 000b because the host controller does not support this optional feature.		0h	RO
07	LHCR	**Light Host Controller Reset:** This optional reset is not supported and is hardwired to 0.		0h	RO
06	IAAD	**Interrupt on Async Advance Doorbell:** This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to *ring* the doorbell. When the host controller has evicted all appropriate cached schedule state, it sets the *Interrupt on Async Advance* status bit in the USBSTS register. If the *Interrupt on Async Advance Enable* bit in the USBINTR register is a one then the host controller will assert an interrupt at the next interrupt threshold. See the *EHCI Specification* for operational details. The host controller sets this bit to a zero after it has set the *Interrupt on Async Advance* status bit in the USBSTS register to a one. Software must not write a one to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.		0h	RW
05	ASY_SCEN	**Asynchronous Schedule Enable:** Default 0b. This bit controls whether the host controller skips processing the Asynchronous Schedule. Values mean: 0 = Do not process the Asynchronous Schedule 1 = Use the ASYNCLISTADDR register to access the Asynchronous Schedule.		0h	RW

Table 26-40. Offset 20h: USB2CMD - USB 2.0 Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 20h **Offset End:** 23h	
Size: 32 bit	**Default:** 00080000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	P_SCEN	**Periodic Schedule Enable:** Default 0b. This bit controls whether the host controller skips processing the Periodic Schedule. Values mean: 0 = Do not process the Periodic Schedule 1 = Use the PERIODICLISTBASE register to access the Periodic Schedule.		0h	RW
03 :02	FLS	**Frame List Size:** Hardwired to 00b because it only supports the 1024-element frame list size.		0h	RO
01	HCRESET	**Host Controller Reset:** This control bit used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset (i.e., RSMRST# assertion and PWROK deassertion). When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. PCI Configuration registers and Host Controller Capability Registers are not affected by this reset. All operational registers, including port registers and port state machines are set to their initial values. Port ownership reverts to the companion host controller(s), with the side effects described in the *EHCI Specification*. Software must reinitialize the host controller in order to return the host controller to an operational state. This bit is set to zero by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Software must not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.		0h	RW
00	RS	**Run / Stop:** Default 0b. 1=Run. 0=Stop. 1 = The Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to a 1. 0 = The Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 microframes after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write a 1 to this field unless the host controller is in the Halted state (i.e., HCHalted in the USBSTS register is a one). The following table explains how the different combinations of Run and Halted must be interpreted: Run/Stop Halted Interpretation 0 0 Valid - in the process of halting 0 1 Valid - halted 1 0 Valid - running 1 1 Invalid - the HCHalted bit clears immediately Memory read cycles initiated by the EHC that receive any status other than Successful will result in this bit being cleared (and also affect the Host Error bit).		0h	RW

26.3.2.2 Offset 24h: USB2STS - USB 2.0 Status Register

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it. See the Interrupts description in section 4 of the *EHCI Specification* for additional information concerning USB 2.0 interrupt conditions.

Table 26-41. Offset 24h: USB2STS - USB 2.0 Status Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** 24h **Offset End:** 27h	
Size: 32 bit	**Default:** 00001000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved.		0h	
15	ASY_SSTAT	**Asynchronous Schedule Status:** This bit reports the current real status of the Asynchronous Schedule. 0 = The status of the Asynchronous Schedule is disabled. 1 = The status of the Asynchronous Schedule is enabled. The Host Controller is not required to *immediately* disable or enable the Asynchronous Schedule when software transitions the *Asynchronous Schedule Enable* bit in the USBCMD register. When this bit and the *Asynchronous Schedule Enable* bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).		0h	RO
14	PER_SSTAT	**Periodic Schedule Status:** This bit reports the current real status of the Periodic Schedule. 0 = The status of the Periodic Schedule is disabled. 1 = The status of the Periodic Schedule is enabled. The Host Controller is not required to *immediately* disable or enable the Periodic Schedule when software transitions the *Periodic Schedule Enable* bit in the USBCMD register. When this bit and the *Periodic Schedule Enable* bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).		0h	RO
13	RECL	**Reclamation:** This is a read-only status bit, which is used to detect an empty asynchronous schedule. The operational model and valid transitions for this bit are described in Section 4 of the *EHCI Specification*.		0h	RO
12	HCH	**HCHalted:** 0 = This bit is a zero whenever the Run/Stop bit is a one. 1 = The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g., internal error).		1h	RO
11 :06	Reserved	Reserved.		0h	
05	INT_ASYA	**Interrupt on Async Advance:** System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the *Interrupt on Async Advance Doorbell* bit in the USBCMD register. This status bit indicates the assertion of that interrupt source.		0h	RWC

Table 26-41. Offset 24h: USB2STS - USB 2.0 Status Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 24h **Offset End:** 27h	
Size: 32 bit	**Default:** 00001000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	HS_ERR	**Host System Error:** 0 = No serious error occurred during a host system access involving the Host Controller module. 1 = The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. Memory read cycles initiated by the EHC that receive any status other than Successful will result in this bit being set. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. A hardware interrupt is generated to the system (if enabled in the Interrupt Enable Register).		0h	RWC
03	FLRO	**Frame List Rollover:** 0 = No *Frame List Index* rollover from its maximum value to 0. 1 = The Host Controller sets this bit to a one when the *Frame List Index* rolls over from its maximum value to zero. Since only 1024-entry Frame List Size is supported, the *Frame List Index* rolls over every time FRNUM[13] toggles.		0h	RWC
02	PCD	**Port Change Detect:** The Host Controller sets this bit to a one when any port for which the *Port Owner* bit is set to zero has a change bit transition from a zero to a one or a Force Port Resume bit transition from a zero to a one as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the *Connect Status Change* being set to a one after system software has relinquished ownership of a connected port by writing a zero to a port's *Port Owner* bit. This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that, on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including Force port resume, overcurrent change, enable/disable change and connect status change). Regardless of the implementation, whenever this bit is readable, i.e., in the D0 state, it must provide a valid view of the Port Status registers.		0h	RWC

Table 26-41. Offset 24h: USB2STS - USB 2.0 Status Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 24h **Offset End:** 27h	
Size: 32 bit	**Default:** 00001000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	USBERRINT	**USB Error Interrupt:** 0 = No error condition. 1 = The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition, e.g., error counter underflow. If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and Bit 0 are set. See the *EHCI Specification* for a list of the USB errors that will result in this interrupt being asserted.		0h	RWC
00	USBINT	**USB Interrupt:** 0 = No completion of a USB transaction whose Transfer Descriptor had its IOC bit set. No short packet is detected. 1 = The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).		0h	RWC

26.3.2.3 Offset 28h: USB2INTR - USB 2.0 Interrupt Enable Register

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Interrupt sources that are disabled in this register still appear in the Status Register to allow the software to poll for events.

Each interrupt enable bit description indicates whether it is dependent on the interrupt threshold mechanism (see Section 4 of the *EHCI Specification*).

Table 26-42. Offset 28h: USB2INTR - USB 2.0 Interrupt Enable Register

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** 28h **Offset End:** 2Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :06	Reserved	Reserved.		0h	
05	INT_AAEN	**Interrupt on Async Advance Enable:** 0 = Disable. 1 = When this bit is a one, and the *Interrupt on Async Advance* bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the *Interrupt on Async Advance* bit.		0h	RW
04	HSE_EN	**Host System Error Enable:** 0 = Disable. 1 = When this bit is a one, and the *Host System Error Status* bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the *Host System Error* bit.		0h	RW
03	FLR_EN	**Frame List Rollover Enable:** 0 = Disable. 1 = When this bit is a one, and the *Frame List Rollover* bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the *Frame List Rollover* bit.		0h	RW
02	PCI_EN	**Port Change Interrupt Enable:** 0 = Disable. 1 = When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the *Port Change Detect* bit.		0h	RW
01	USBEI_EN	**USB Error Interrupt Enable:** 0 = Disable. 1 = When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software by clearing the *USBERRINT* bit.		0h	RW
00	USBI_EN	**USB Interrupt Enable:** 0 = Disable. 1 = When this bit is a one, and the USBINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software by clearing the *USBINT* bit.		0h	RW

Note: For all enable register bits, 1= Enabled, 0= Disabled

26.3.2.4 Offset 2Ch: FRINDEX - Frame Index Register

This register is used by the host controller to index into the periodic frame list. The register updates every 125 microseconds (once each microframe). Bits [12:3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index is fixed at 10 since only 1024-entry frame lists are supported. This register must be written as a DWord. Word and byte writes produce undefined results. This register cannot be written unless the Host Controller is in the Halted state as indicated by the *HCHalted* bit (USB 2.0STS register). A write to this register while the Run/Stop bit is set to a one (USB 2.0CMD register) produces undefined results. Writes to this register also affect the SOF value. See Section 4 of the *EHCI Specification* for details.

Table 26-43. Offset 2Ch: FRINDEX - Frame Index Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 2Ch **Offset End:** 2Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :14	Reserved	Reserved.		0h	
13 :00	FLCI	**Frame List Current Index/Frame Number:** The value in this register increments at the end of each time frame (e.g., microframe).Bits [12:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or microframes) before moving to the next index.		0h	RW

The SOF frame number value for the bus SOF token is derived or alternatively managed from this register. Please refer to Section 4 of the *EHCI Specification* for a detailed explanation of the SOF value management requirements on the host controller. The value of FRINDEX must be 125 µs (1 microframe) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is 11 bits and is named SOFV. SOFV updates every 8 microframes. (1 ms). An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from a zero to a one.

Software must use the value of FRINDEX to derive the current microframe number and to provide the *get* microframe number function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if chip is reset or software writes to FRINDEX. Writes to FRINDEX must also *write-through* FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software must never write a FRINDEX value where the three least significant bits are 111b or 000b.

26.3.2.5 Offset 30h: CTRLDSSEGMENT - Control Data Structure Segment Register

This 32-bit register corresponds to the most significant address bits [63:32] for all EHCI data structures. Since the 64-bit Addressing Capability field is hardwired in HCCPARAMS to one, then this register is used with the link pointers to construct 64-bit addresses to EHCI control data structures. This register is concatenated with the link pointer from either the PERIODICLISTBASE, ASYNCLISTADDR, or any control data structure link field to construct a 64-bit address. This register allows the host software to locate all control data structures within the same 4 GByte memory segment.

Note: CMIhas 44-bit addressing internally for both control and data structures. Address bits 63:44 [31:12] are hardwired to zero independent of the setting of this register. The lower 12 address bits 43:32 [11:0] are fully read/write capable for software compatibility and specification compliance.

Table 26-44. Offset 30h: CTRLDSSEGMENT - Control Data Structure Segment Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 30h **Offset End:** 33h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :12	HC_UP	**Upper Address[63:44]:** This 20-bit field is hard wired to zero.		0h	RO
11 :00	HC_LWR	**Upper Address[43:32]:** This 12-bit field corresponds to address bits 43:32 when forming a control data structure address.		0h	RW

26.3.2.6 Offset 34h: PERIODICLISTBASE - Periodic Frame List Base Address Register

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. Since the host controller operates in 64-bit mode (as indicated by the one in the 64-bit Addressing Capability field in the HCCSPARAMS register) (offset 08h, bit 0), then the most significant 32 bits of every control data structure address comes from the CTRLDSSEGMENT register. System software loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4 Kbyte aligned. The contents of this register are combined with the Frame Index Register (FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Table 26-45. Offset 34h: PERIODICLISTBASE - Periodic Frame List Base Address Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 34h **Offset End:** 37h	
Size: 32 bit	**Default:** 00000XXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :12	HC_LOW	**Base Address (Low):** These bits correspond to memory address signals [31:12], respectively.		0h	RW
11 :00	Reserved	Reserved. Must be written as 0s. During runtime, the value of these bits are undefined.		XXXh	

26.3.2.7 Offset 38h: ASYNCLISTADDR - Current Asynchronous List Address Register

This 32-bit register contains the address of the next asynchronous queue head to be executed. Since the host controller operates in 64-bit mode (as indicated by a one in 64-bit Addressing Capability field in the HCCPARAMS register), then the most significant 32 bits of every control data structure address comes from the CTRLDSSEGMENT register. Bits [4:0] of this register cannot be modified by system software and will always return zeros when read. The memory structure referenced by this physical memory pointer is assumed to be 32-byte aligned.

Table 26-46. Offset 38h: ASYNCLISTADDR - Current Asynchronous List Address Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 38h **Offset End:** 3Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :05	LPL	**Link Pointer Low:** These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).		0h	RW
04 :00	Reserved	Reserved.		0h	

26.3.2.8 Offset 60h: CONFIGFLAG - Configure Flag Register

This register is in the suspend power well. It is only reset by hardware when the suspend power is initially applied or in response to a host controller reset. This 32-bit register contains the address of the next asynchronous queue head to be executed. Since the host controller operates in 64-bit mode (as indicated by a 1 in 64-bit Addressing Capability field in the HCCPARAMS register), then the most significant 32 bits of every control data structure address comes from the CTRLDSSEGMENT register (offset 08h). Bits [4:0] of this register cannot be modified by system software and will always return 0's when read. The memory structure referenced by this physical memory pointer is assumed to be 32-byte aligned.

Table 26-47. Offset 60h: CONFIGFLAG - Configure Flag Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 60h **Offset End:** 63h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :01	Reserved	Reserved. Reads from this field will always return 0.		0h	
00	CF	**Configure Flag:** Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. Bit values and side effects are listed below. See section 4 of the *EHCI Specification* for operation details. 0 = Port routing control logic default-routes each port to the classic host controllers. 1 = Port routing control logic default-routes all ports to this host controller.		0h	RW

26.3.2.9 Offset 64h: PORTSC - Port *N* Status and Control Register

A host controller must implement one or more port registers. Software uses the N_Port information from the Structural Parameters Register to determine how many ports need to be serviced. All ports have the structure defined below. Software must not write to unreported Port Status and Control Registers.

This register is in the suspend power well. It is only reset by hardware when the suspend power is initially applied or in response to a host controller reset. The initial conditions of a port are:

- No device connected
- Port disabled

Note: When a device is attached, the port state transitions to the connected state and system software will process this as with any status change notification. Refer to Section 4 of the *EHCI Specification* for operational requirements for how change events interact with port suspend mode.

Note: If a port is being used as the Debug Port, then the port may report device connected and enabled when the Configured Flag is a zero.

Note: There is no support for wake from USB when in S3/S4/S5.

Table 26-48. Offset 64h: PORTSC - Port N Status and Control Register (Sheet 1 of 5)

Description: Port 1 64 - 67h, Port 2 68 - 6Bh					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 64h **Offset End:** 67h	
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00003000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :23	Reserved	Reserved.		000h	
22	WKOC_E	**Wake on Overcurrent Enable:** 0 = Disable. (Default). 1 = Writing this bit to a one enables the port to be sensitive to overcurrent conditions as wake-up events. When enabled to do so, the EHC sets the PME Status bit in the Power Management Control/Status Register (offset 54, bit 15) when the overcurrent Active bit (bit 4 of this register) is set. Note: There is no support for wake from USB when in S3/S4/S5.		0h	RW
21	WKDSCNNT_E	**Wake on Disconnect Enable:** 0 = Disable. (Default). 1 = Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events. When enabled to do so, the EHC sets the PME Status bit in the Power Management Control/Status Register (offset 54, bit 15) when the Current Connect Status changes from connected to disconnected (i.e., bit 0 of this register changes from 1 to 0). Note: There is no support for wake from USB when in S3/S4/S5.		0h	RW

Table 26-48. Offset 64h: PORTSC - Port N Status and Control Register (Sheet 2 of 5)

Description: Port 1 64 - 67h, Port 2 68 - 6Bh					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 64h **Offset End:** 67h	
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00003000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
20	WKCNNT_E	**Wake on Connect Enable:** 0 = Disable. (Default). 1 = Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. When enabled to do so, the EHC sets the PME Status bit in the Power Management Control/Status Register (offset 54, bit 15) when the Current Connect Status changes from disconnected to connected (i.e., bit 0 of this register changes from 0 to 1). Note: There is no support for wake from USB when in S3/S4/S5.		0h	RW
19 :16	PT_CTRL	**Port Test Control:** Default = 0000b. When this field is zero, the port is NOT operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by the specific value. The encoding of the test mode bits are (0110b - 1111b are reserved): Bits Test Mode 0000b Test mode not enabled 0001b Test J_STATE - During this test mode the hardware will force pre-emphasis disabled to the AFE 0010b Test K_STATE - During this test mode the hardware will force pre-emphasis disabled to the AFE 0011b Test SE0_NAK 0100b Test Packet 0101b Test FORCE_ENABLE Refer to *USB Rev. 2.0 Specification*, Chapter 7 and the *EHCI Specification*, Chapter 4 for details on each test mode. The EHC does not support the option to run the port tests while the Run/Stop bit is a one.		0000h	RW
15 :14	**Reserved**	**Reserved.**		00h	
13	PO	**Port Owner:** Default = 1b. This bit unconditionally goes to a 0b when the *Configure Flag* makes a 0b to 1b transition. This bit unconditionally goes to 1b whenever the *Configure Flag* bit is zero. System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that a companion host controller owns and controls the port. See Section 4 of the *EHCI Specification* for operational details.		1h	RW
12	PP	**Port Power:** Hard-wired with a value of one. This indicates that the port does have power.		1h	RO

Table 26-48. Offset 64h: PORTSC - Port N Status and Control Register (Sheet 3 of 5)

Description: Port 1 64 - 67h, Port 2 68 - 6Bh					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 64h **Offset End:** 67h	
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00003000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 :10	LS	**Line Status:** These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is zero and the current connect status bit is set to a one. The encoding of the bits is as follows: Bits[11:10] / USB State / Interpretation 00b / SE0 / Not Low-speed device, perform EHCI reset 10b / J-state / Not Low-speed device, perform EHCI reset 01b / K-state / Low-speed device, release ownership of port 11b / Undefined / Not Low-speed device, perform EHCI reset		0h	RO
09	Reserved	Reserved.		0h	
08	PR	**Port Reset:** 0 = Port is not in Reset (default). 1 = Port is in Reset. When software writes a one to this bit (from a zero), the bus reset sequence as defined in the *USB Rev. 2.0 Specification* is started. Software writes a zero to this bit to terminate the bus reset sequence. Software must keep this bit at a one long enough to guarantee the reset sequence, as specified in the *USB Rev. 2.0 Specification*, completes. ***Note:*** When software writes this bit to a one, it must also write a zero to the *Port Enable* bit. ***Note:*** When software writes a zero to this bit, there may be a delay before the bit status changes to a zero. The bit status will not read as a zero until after the reset has completed. If the port is in high-speed mode after reset is complete, the host controller will automatically enable this port (e.g., set the *Port Enable* bit to a one). A host controller must terminate the reset and stabilize the state of the port within 2 ms of software transitioning this bit from a one to a zero. For example, if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2 ms of software writing this bit to a zero. The HCHalted bit in the USB2STS register must be a zero before software attempts to use this bit. The host controller may hold Port Reset asserted to a one when the HCHALTED bit is a one. The *Run/Stop* bit in the Command Register must be set in order for the *Port Reset* bit to be cleared.		0h	RW

Table 26-48. Offset 64h: PORTSC - Port N Status and Control Register (Sheet 4 of 5)

Description: Port 1 64 - 67h, Port 2 68 - 6Bh					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 64h **Offset End:** 67h	
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00003000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	PS	**Suspend:** 0 = Port not in suspend state (default). 1 = Port in suspend state. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0X — Disable 10 — Enable 11 — Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to a 1. In the suspend state, the port is sensitive to resume detection. The bit status does not change until the port is suspended and there may be a delay in suspending a port if there is a transaction currently in progress on the USB. A write of zero to this bit is ignored by the host controller. The host controller will unconditionally set this bit to a zero when: • Software sets the *Force Port Resume* bit to a zero (from a one). • Software sets the *Port Reset* bit to a one (from a zero). If host software sets this bit to a one when the port is not enabled (i.e., Port enabled bit is a zero) the results are undefined.		0h	RW
06	FPR	**Force Port Resume:** 0 = No resume (K-state) detected/driven on port (default). 1 = Resume detected/driven on port. This functionality defined for manipulating this bit depends on the value of the *Suspend* bit. For example, if the port is not suspended (*Suspend* and *Enabled* bits are a one) and software transitions this bit to a one, then the effects on the bus are undefined. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the *Port Change Detect* bit in the USBSTS register is also set to a one. If software sets this bit to a one, the host controller must not set the *Port Change Detect* bit. When the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the *USB Rev. 2.0 Specification*. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. Software must appropriately time the Resume and set this bit to a zero when the appropriate amount of time has elapsed. Writing a zero (from one) causes the port to return to high-speed mode (forcing the bus below the port into a high-speed idle). This bit will remain a one until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to a zero.		0h	RW
05	OCC	**Overcurrent Change:** 0 = No change (default). 1 = This bit gets set to a one when there is a change to the Overcurrent Active bit. Software clears this bit by writing a one to this bit position. The functionality of this bit is not dependent upon the port owner.		0h	RWC

Table 26-48. Offset 64h: PORTSC - Port N Status and Control Register (Sheet 5 of 5)

Description: Port 1 64 - 67h, Port 2 68 - 6Bh					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 64h **Offset End:** 67h	
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** 68h **Offset End:** 6Bh	
Size: 32 bit	**Default:** 00003000h			**Power Well:** Suspend	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	OCA	**Overcurrent Active:** 0 = This port does not have an overcurrent condition (deafult). 1 = This port currently has an overcurrent condition. This bit will automatically transition from a one to a zero when the overcurrent condition is removed. The functionality of this bit is not dependent upon the port owner. CMI automatically disables the port when the overcurrent active bit is '1'.		0h	RO
03	PEDC	**Port Enable / Disable Change:** 0 = No change (default). 1 = Port enabled/disabled status has changed. For the root hub, this bit gets set to a one only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the *USB Specification* for the definition of a port error). This bit is not set due to the Disabled-to-Enabled transition, nor due to a disconnect. Software clears this bit by writing a 1 to it.		0h	RWC
02	PENDIS	**Port Enabled / Disabled:** 0 = Disable (default). 1 = Enable. As described in the *EHCI Specification*, ports are enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. The host controller will only set this bit to a one when the reset sequence determines that the attached device is a high-speed device.		0h	RW
01	CSC	**Connect Status Change:** 0 = No change (default). 1 = Change in Current Connect Status. Indicates a change has occurred in the port's Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit will remain set). Software sets this bit to 0 by writing a 1 to it.		0h	RWC
00	CCS	**Current Connect Status:** 0 = No device is present (default). 1 = Device is present on port. This value reflects the current state of the port and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.		0h	RO

26.4 EHC Initialization

The following describes the expected EHC initialization sequence in chronological order, beginning with a complete power cycle in which the suspend well and core well have been off.

26.4.1 Power On

The suspend well is a “deeper” power plane than the core well, which means that the suspend well is always functional when the core well is functional but the core well may not be functional when the suspend well is. Therefore, the suspend well reset pin (RSMRST#) deasserts before the core well reset pin (PWROK) rises.

1. The suspend well reset deasserts, leaving all registers and logic in the suspend well in the default state. However, it is not possible to read any registers until after the core well reset deasserts.
2. The core well reset deasserts, leaving all registers and logic in the core well in the default state. The EHC configuration space is accessible at this point. The core well reset can (and typically does) occur without the suspend well reset asserting. This means that all of the Configure Flag and Port Status and Control bits (and any other suspend-well logic) may be in any valid state at this time.

26.4.2 Driver Initialization

See Chapter 4 of the *EHCI Specification, Rev. 1.0*.

26.4.3 EHC Resets

In addition to the standard hardware resets, portions of the EHC are reset by the HCRESET bit and the transition from the D3hot device power management state to the D0 state. The effect of each of these resets are:

Table 26-49. HCRESET Bit Summary

Reset	Does Reset	Does Not Reset	Comments
HCRESET bit set	Memory space registers except Structural Parameters (which is written by BIOS)	Configuration Registers	The HCRESET must only affect registers that the EHCI driver controls. PCI Configuration space and BIOS-programmed parameters must not be reset.
Software writes the Device Power State from D3hot (11b) to D0 (00b)	Core-well registers (except BIOS-programmed registers)	Suspend-well registers; BIOS-programmed Core-well Registers	The D3-to-D0 transition must not cause wake information (suspend well) to be lost. It also must not clear BIOS-programmed registers because BIOS may not be invoked following the D3-to-D0 transition.

If the detailed register descriptions give exceptions to these rules, those exceptions override these rules. This summary is provided to help explain the reasons for the reset policies.

26.5 Data Structures in Main Memory

See Section 3 and Appendix B of the *EHCI Specification, Rev. 1.0* for details.

26.6 USB 2.0 Enhanced Host Controller DMA

The USB 2.0 enhanced host controller implements three sources of USB packets. They are, in order of priority on USB during each microframe,

1. the USB 2.0 Debug Port (see Section 26.13),
2. the Periodic DMA engine, and
3. the Asynchronous DMA engine.

CMI always performs any currently-pending debug port transaction at the beginning of a microframe, followed by any pending periodic traffic for the current microframe. If there is time left in the microframe, then the EHC performs any pending asynchronous traffic at the end of the microframe (EOF1). The debug port traffic is only presented on one port (Port #0), while the other ports are idle during this time.

The following subsections describe the policies of the periodic and asynchronous DMA engines.

26.6.1 Periodic List Execution

The Periodic DMA engine contains buffering for two control structures (two transactions). By implementing two entries, the EHC is able to pipeline the memory accesses for the next transaction while executing the current transaction on the USB ports. A multiple-packet, high-bandwidth transaction occupies one of these buffer entries, which means that up to six 1 Kbyte data packets may be associated with the two buffered control structures.

In order to simplify the pipelined implementation that is optimized for normal execution, the EHC does not implement immediate retries on High Bandwidth Interrupt transactions that encounter transaction errors (for ins and outs) or a Data Toggle mismatch (for Interrupt In). This is an optional implementation, but not recommended, by the *USB Specification* and the *USB Specification, Revision 0.95.* The EHC will reattempt the transaction when that qTD is encountered again in the periodic schedule. If successful when reattempted, then the EHC will continue with the multiple packets allowed by the high-bandwidth endpoint during that same microframe.

26.6.1.1 Read Policies for Periodic DMA

The Periodic DMA engine performs memory reads for the following structures:

Table 26-50. Periodic DMA Engine Memory Reads

Memory Structure	Size (DWORDS)	Comments
Periodic Frame List entry	1	The EHC reads the entry for each microframe. The frame list is not internally cached across microframes.
Frame Span Traversal Node	2	
iTD	23	Only the 64-bit addressing format is supported.
siTD	9	Only the 64-bit addressing format is supported.

Table 26-50. Periodic DMA Engine Memory Reads

qTD	13	Only the 64-bit addressing format is supported.
Queue Head	17	Only the 64-bit addressing format is supported.
Out Data	Up to 257	Large read requests are broken down into smaller aligned read requests based on the setting of the Read Request Maximum Length field.

Periodic DMA read policies:

1. The EHC Periodic DMA Engine (PDE) does not generate accesses to main memory unless all three of the following conditions are met:
   a. The HCHalted bit is 0 (memory space, offset 24h, bit 12). Software clears this bit indirectly by setting the RUN/STOP bit to 1.
   b. The Periodic Schedule Status bit is 1 (memory space, offset 24h, bit 14). Software sets this bit indirectly by setting the Periodic Schedule Enable Bit to 1.
   c. The Bus Master Enable bit is 1 (configuration space, offset 04h, bit 2).
2. Once the above conditions are met, the PDE waits until the frame index counter rolls over from the end of microframe 6 to the beginning of microframe 7 to begin prefetching for microframe 0 of the next frame. This means the initial memory access may be delayed up to 1 ms after the DMA-enabled conditions are met. Further delays within the arbitration and datapath are also possible before the first read request is presented on the IMCH/IICH link.
3. The Periodic Frame List Entry is always read from memory before any data structures associated with the new microframe are accessed.
4. Prefetching is limited to the current and next microframes only. If prefetching is disabled, the periodic DMA engine will perform transactions serially (no pipelining) and will read structures for the current microframe only.
5. The PDE fetches structures in the periodic list until all information (including data) is available to run one USB transaction before beginning to fetch the structures for a pipelined transaction. For High-Bandwidth Out transactions, all of the data may not fit into the Data FIFO; in those cases, the next pipelined control structure fetches will be delayed until some data is delivered to USB.
6. The PDE does not refetch the control structure between “Multi” packets of a High Bandwidth endpoint.
7. The PDE will not generate any control structure reads (including the frame list index) if both of the transaction buffers are occupied. Data reads are the only read requests that will be generated by the PDE in this case.
8. The PDE will not pipeline fetch a control structure (iTD, siTD, or QH) if the other transaction slot contains that control structure already. This is to avoid executing based on stale fields in the control structure since a status write (or overlay) is expected to occur following execution of the pending transaction. The PDE will traverse the schedule (periodic frame list entry and any inactive control structures for the microframe) before encountering the Link Pointer to the stale structure. At that point the fetching pauses until the pending transaction is completed. *The iTD could be refetched since a separate status is maintained for each microframe; the PDE will not attempt this optimization*.
9. Once the PDE checks the length of a periodic packet against the remaining time in the microframe (late-start check) and decides that there is not enough time to run it on the wire, then the EHC switches over to run asynchronous traffic. The EHC does not attempt to look for any shorter packets in the remainder of the periodic schedule that might be able to fit in the current microframe.

10. The PDE implements a “Gross Late-Start” check which determines whether any more control or data structure reads will be initiated for transactions associated with the current microframe. The threshold for this check is determined by the Gross Late Start Cut-Off field in configuration register offset 84h.
11. An entry in the 2-deep command FIFO becomes available for a new transaction fetch when any of the following events occur:
    a. The final transaction results are posted in write buffers to memory.
    b. Either of the late-start checks fail for this transaction (or the preceding transaction in the same microframe).
    c. A High-Bandwidth Interrupt transaction times out. For High-Bandwidth Interrupt transactions that time out, the CMI does not immediately retry the transaction as recommended by the *USB Specification* (Section 5.9.1). Instead, all control and data structures are flushed and the transaction is reattempted the next time that endpoint is scheduled.

Note: When a host error occurs, the commands are kept in the PDE. The EHCI software driver must assert the HCRESET in order to clear the pending transactions before reenabling the PDE.

12. Data fetches are not initiated unless there is room in the Out Data FIFO to consume the amount of data requested.
13. Read requests are broken up and throttled based on the Read Request Maximum Length field and the Request Rate Throttle fields in the configuration register at offset FCh. Control or Data structures that cross a Maximum Length-aligned boundary in memory are broken into multiple requests. This allows other packets from within the IICH to be interleaved on the IMCH/IICH link and through the memory controller to avoid temporary starvation of those functions. When generating the multiple read requests, the EHC will naturally-align the requests (i.e., 64-byte requests will not fetch across 64-byte address boundaries in memory). This guarantees that, as cache-line sizes increase, the back-to-back requests do not cause double-snoops on specific cache lines. Unlike control structure read requests, only reads for data will be subject to the Request Rate Throttle.
14. Asynchronous DMA memory accesses may be interleaved at any point with the periodic DMA memory accesses on the IMCH/IICH link.

26.6.1.2 Write Policies for Periodic DMA

The Periodic DMA engine performs writes to the following data structures:

Periodic DMA write policies:

1. The Periodic DMA Engine (PDE) will only generate writes after a transaction is executed on USB. Some important notes associated with this rule are:
    a. If either of the late-start checks fails before the transaction is run on the USB ports, then none of the writes normally associated with that transaction will occur. High-Bandwidth Exception: If the late-start check fails after the first packet of a High-Bandwidth (multi) transaction is executed but before the last packet, then the PDE must write the status for any completed transfers to memory.
    b. The Queue Head Overlay write occurs after the first transaction for a qTD is completed on the USB interface.
2. Status writes are always performed after In Data writes for the same transaction.
3. When writing the status back to the two siTDs associated with a backpointer, the PDE first writes to the siTD which was referenced by the backpointer and secondly writes to the siTD which contains the backpointer.

4. Asynchronous DMA memory accesses may be interleaved at any point with the periodic DMA memory accesses on the IMCH/IICH link.
5. When writing back the qTD information after clearing the *Active* bit, the *EHCI Specification* does not require that the *C_Page* field is written. However, due to byte-granular write control, the EHC does write to this field, and the value is not necessarily the final or incremented *C_Page* value.

26.6.2 Asynchronous List Execution

The Asynchronous DMA engine contains buffering for two control structures (two transactions). By implementing two entries, the EHC is able to pipeline the memory accesses for the next transaction while executing the current transaction on the USB ports.

26.6.2.1 Read Policies for Asynchronous DMA

The Asynchronous DMA engine performs reads for the following structures:

Table 26-51. Asynchronous DMA Engine Reads

Memory Structure	Size (DWORDS)	Comments
qTD	13	Only the 64-bit addressing format is supported.
Queue Head	17	Only the 64-bit addressing format is supported.
Out Data	Up to 129	Large read requests are broken down in to smaller aligned read requests based on the setting of the Read Request Maximum Length field.

Asynchronous DMA read policies:

1. The EHC Asynchronous DMA Engine (ADE) does not generate accesses to main memory unless all four of the following conditions are met. (The ADE may be active when the periodic schedule is actively executed, unlike the description in the *EHCI Specification*; since the EHC contains independent DMA engines, the ADE may perform memory accesses interleaved with the PDE accesses.)
   a. The HCHalted bit is 0 (memory space, offset 24h, bit 12). Software clears this bit indirectly by setting the RUN/STOP bit to 1.
   b. The Asynchronous Schedule Status bit is 1 (memory space, offset 24h, bit 14). Software sets this bit indirectly by setting the Asynchronous Schedule Enable Bit to 1.
   c. The Bus Master Enable bit is 1 (configuration space, offset 04h, bit 2).
   d. The ADE is not sleeping due to the detection of an empty schedule. There is not one single bit that indicates this state. However, the sleeping state is entered when the Queue Head with the H bit set is encountered when the Reclamation bit in the USB 2.0 Status register is 0.
2. Once the above conditions are met, the ADE immediately begins reading the Queue Head to which the Current Asynchronous List Address Register points. Delays within the arbitration and datapath are possible before the first read request is presented on the IMCH/IICH link.
3. If prefetching is disabled, the ADE will perform transactions serially (no pipelining).
4. The ADE fetches structures in the asynchronous list until all information (including data) is available to run one USB transaction before beginning to fetch the structures for a pipelined transaction.
5. The ADE will not generate any control structure reads if both of the transaction buffers are occupied. Data reads are the only read requests that will be generated by the ADE in this case.

6. The ADE does not fetch data when a QH is encountered in the Ping state. An Ack handshake in response to the Ping results in the ADE writing the QH to the Out state, which results in the fetching and delivery of the Out Data on the next iteration through the asynchronous list.
7. The ADE will not pipeline fetch a Queue Head if the other transaction slot contains that Queue Head already (i.e., only one active QH). This is to avoid executing based on stale fields in the Queue Head since a status write (or overlay) is expected to occur following execution of the pending transaction. The ADE will traverse the schedule (any inactive Queue Heads) before encountering the Link Pointer to the stale structure. At that point the fetching pauses until the pending transaction is completed.
8. Once the ADE checks the length of an asynchronous packet against the remaining time in the microframe (late-start check) and decides that there is not enough time to run it on the wire, then the EHC stops all activity on the USB ports for the remainder of that microframe. The EHC does not attempt to look for any shorter packets in the remainder of the asynchronous schedule that might be able to fit in the current microframe. Unlike the PDE, the ADE keeps the transaction internally for executing in the next microframe without refetching from memory.
9. An entry in the 2-deep command FIFO becomes available for a new transaction fetch when any of the following events occur:
    a. The final transaction results are posted in write buffers to memory.
    b. A host error causes an unexpected halt. Any unexecuted transactions in the command FIFO are flushed.
10. Once the ADE detects an “empty” asynchronous schedule as described in the *EHCI Specification*, it implements a waking mechanism like the one in the example. The amount of time that the ADE “sleeps” is 10 µs +/-30 ns.
11. Data fetches are not initiated unless there is room in the Out Data FIFO for the amount of data requested.
12. Read requests are broken up and throttled based on the Read Request Maximum Length field and the Request Rate Throttle fields in the configuration register at offset FCh. Control or Data structures that cross a Maximum Length-aligned boundary in memory are broken into multiple requests. This allows other packets from within the IICH to be interleaved on the IMCH/IICH link and through the memory controller to avoid temporary starvation of those functions. When generating the multiple read requests, the EHC will naturally-align the requests (i.e., 64-byte requests will not fetch across 64-byte address boundaries in memory). This guarantees that, as cache-line sizes increase, the back-to-back requests do not cause double-snoops on specific cache lines. Unlike control structure read requests, only reads for data will be subject to the Request Rate Throttle.
13. Periodic DMA memory accesses may be interleaved at any point with the asynchronous DMA memory accesses on IMCH/IICH link.

26.6.2.2 Write Policies for Asynchronous DMA

The Asynchronous DMA engine performs writes to the following memory structures:

Table 26-52. Asynchronous DMA Engine Writes

Memory Structure	Size (DWORDS)	Comments
Asynchronous Queue Head Overlay	14	Only the 64-bit addressing format is supported. Dwords 0Ch through 43h are written.
Asynchronous Queue Head Status Write	3	Dwords 14h through 1Fh are written.
Asynchronous qTD Status Write	3	Dwords 04h through 0Fh are written. PID Code, IOC, Buffer Pointer (page 0), and Alt. Next qTD Pointer are rewritten with the original value.
In Data	Up to 129	Data writes are broken down into 16 Dword-aligned chunks.

Asynchronous DMA write policies:

1. The Asynchronous DMA Engine (ADE) will only generate writes after a transaction is executed on USB. Some important notes associated with this rule are:
   a. If the late-start check fails before the transaction is run on the USB ports, then the USB transaction and the writes are delayed until the next opportunity to run the asynchronous traffic.
   b. The Queue Head Overlay write occurs after the first transaction for a qTD is completed on the USB interface.
2. Status writes are always performed after In Data writes for the same transaction.
3. Periodic DMA memory accesses may be interleaved at any point with the Asynchronous DMA memory accesses on IMCH/IICH link.
4. When writing back the qTD information after clearing the *Active* bit, the *EHCI Specification* does not require that the *C_Page* field is written. However, due to byte-granular write control, the EHC does write to this field, and the value is not necessarily the final or incremented *C_Page* value.

26.7 Data Encoding and Bit Stuffing

See the *USB Rev. 2.0 Specification.*

26.8 Packet Formats

See the *USB Rev. 2.0 Specification.*

26.9 USB 2.0 Interrupts and Error Conditions

The *EHCI Specification* goes into detail on the EHC interrupts and the error conditions that cause them. All error conditions that the EHC detects can be reported through the EHCI Interrupt status bits. Only CMI-specific interrupt and error-reporting behavior is documented in this section. The EHCI Interrupts Section (in the *EHCI Specification*) must be read first, followed by this section, to fully comprehend the EHC interrupt and error-reporting functionality.

- Based on the EHC's Buffer sizes and buffer management policies, the Data Buffer Error can never occur.
- Master Abort and Target Abort responses from the system interface on EHC-initiated read packets will be treated as Fatal Host Errors. The EHC halts when these conditions are encountered.

- CMI may assert the interrupts that are based on the interrupt threshold as soon as the status for the last complete transaction in the interrupt interval has been posted in the internal write buffers. The requirement in the *EHCI Specification* (that the status is written to memory) is met internally, even though the write may not be seen on the IMCH/IICH interface before the interrupt is asserted.
- Since CMI only supports the 1024-element Frame List size, the Frame List Rollover interrupt occurs every 1024 milliseconds.
- CMI delivers interrupts using PIRQ#[A].
- CMI does not modify the CERR count on an Interrupt IN when the “Do Complete-Split” execution criteria are not met.
- For complete-split transactions in the Periodic list, the “Missed Microframe” bit does not get set on a control-structure-fetch that fails the late-start test. If subsequent accesses to that control structure do not fail the late-start test, then the “Missed Microframe” bit will get set and written back.

26.9.1 Aborts on USB 2.0-Initiated Memory Reads

If a read initiated by the EHC receives any status other than “Successful” in the completion packet, the EHC treats it as a fatal host error. The following actions are taken when this occurs:

- The Host System Error status bit is set
- The DMA engines are halted, the Run/Stop bit is cleared, and the HCHalted bit is set, after completing up to one more transaction on the USB interface
- If enabled (by the Host System Error Enable), then an interrupt is generated
- If the status is Master Abort, then the Received Master Abort bit in configuration space is set
- If the status is Target Abort, then the Received Target Abort bit in configuration space is set
- If enabled (by the SERR Enable bit and the SERR on Abort Enable bit in the function’s configuration space), then the Signaled System Error bit in configuration bit is set and the internal SERR signal is asserted

26.9.2 Host Interface Parity Errors

In the event of parity errors on the host-side interface, the EHC is required to respond as shown in the following table.

The EHC is accessible as a target after the parity errors are detected (assuming that

Table 26-53. Host Interface Parity Errors (Sheet 1 of 2)

Input Scenario			Resulting Behavior			
Event	Parity Error Resp	SERR# En (CMD register, bit 8)	DPE (DSR register, bit 15)	Master DPE (DSR register, bit 8)	Host System Error (USB Status)	Notes
Downbound Request Command Parity Error	0	X	1	0	0	Do take the cycle, as normal.
	1	0	1	0	1	Do not take the cycle (master abort). No SERR# generated
	1	1	1	0	1	Do not take the cycle (master abort). SERR# generated

Table 26-53. Host Interface Parity Errors (Sheet 2 of 2)

Input Scenario			Resulting Behavior			
Event	**Parity Error Resp**	**SERR# En (CMD register, bit 8)**	**DPE (DSR register, bit 15)**	**Master DPE (DSR register, bit 8)**	**Host System Error (USB Status)**	**Notes**
Downbound Request Address Parity Error	Identical to the Command Parity Error rows above					
Downbound Request Data Parity Error	0	X	1	0	0	Do take the cycle, as normal.
	1	0	1	0	1	Cycle is taken. Halt the Host Controller, if currently not halted. Drop the write data. No SERR# generated. (No PERR# on the IMCH/IICH link) Software can only determine that the error occurred through the Host Error interrupt or by polling.
	1	1	1	0	1	Cycle is taken. Halt the Host Controller, if currently not halted. Drop the write data. SERR# generated. (No PERR# on the IMCH/IICH link)
Downbound Completion Command Parity Error	This must be treated the same as the Downbound Request Command Parity Error because the error could be on the completion/request bit.					
Downbound Completion Data Parity Error	0	X	1	0	0	Do take the cycle, as normal.
	1	0	1	1	1	Cycle is taken. Halt the Host Controller, if currently not halted. Do not forward data to the USB ports. No SERR# generated. (No PERR# on the IMCH/IICH link) Software can only determine that the error occurred through the Host Error interrupt or by polling.
	1	1	1	1	1	Cycle is taken. Halt the Host Controller, if currently not halted. Do not forward data to the USB ports. SERR# generated. (No PERR# on the IMCH/IICH link)

the system instability has not caused a deadlock in some way).

All of the above description is required behavior. The following text describes the CMI-specific implementation details:

There are three general forms of parity errors that the EHC may detect on its system interface:

- Command/Address (on cycles from the host side)
  - — C/A parity error on Host-initiated cycles, or
  - — Command parity error on Completion packets to CMI

- Host-Initiated Write Data
- Read Completion Data to the EHC

When any of these three errors are detected and the Parity Error Response bit is set in the USB 2.0 function, the EHC immediately sets the Host Error bit to '1' and clears the Run/Stop bit to '0'. At most, one more packet completes on the high-speed USB ports after this occurs (this packet will not contain, or be based upon, the data containing the host error). When that packet is completed, the HCHalted bit is set to a '1'. Once the Host Error bit has been set, the Run/Stop bit can not be set by software until after the HCRESET is generated by software and completed by the EHC, which lasts for multiple milliseconds. The EHC will still accept host-initiated cycles as a target after the HCHalted bit has been set.

It is recommended that software reboot in the event of a parity error because the error could be an indication of other system hardware problems.

26.10 USB 2.0 Power Management

26.10.1 Pause Feature

This feature allows platforms to dynamically enter low-power states during brief periods when the system is idle (i.e., between keystrokes). This is useful for enabling power management features like Enhanced Intel SpeedStep Technology (EIST). The policies for entering these states typically are based on the recent history of system bus activity to incrementally enter deeper power management states. Normally, when the EHC is enabled, it regularly accesses main memory while traversing the DMA schedules looking for work to do; this activity is viewed by the power management software as a non-idle system, thus preventing the power managed states to be entered. Suspending all of the enabled ports can prevent the memory accesses from occurring, but there is an inherent latency overhead with entering and exiting the suspended state on the USB ports that makes this unacceptable for the purpose of dynamic power management. As a result, the EHCI software drivers are allowed to pause the EHC's DMA engines when it knows that the traffic patterns of the attached devices can afford the delay. The pause only prevents the EHC from generating memory accesses; the SOF packets continue to be generated on the USB ports (unlike the suspended state).

The expected sequence of events for the Pause Feature is:

1. When starting the DMA engines for the first time, the enable bits are set at the same time as, or after, the Run bit is set. However, the EHC should be capable of handling the Run bit set to 0 while one or both of the enable bits are 1; this may happen, for example, when the hardware halts the DMA due to an error. The enable bits may be set to 1 by different writes to the Command Register. The EHC takes the following actions when the enable bits are set by software:
   a. The corresponding Asynch/Periodic Schedule Status bit(s) is (are) immediately set to 1.
   b. If the Asynch Enable bit is set, the first queue head in the asynchronous schedule is immediately fetched (if the Bus Master Enable bit in Configuration space is set).
   c. If the Periodic Enable bit is set, then the periodic frame list entry is fetched (if the Bus Master Enable bit in Configuration space is set) on the next internal trigger point, which may be up to 1 ms later.
2. Before clearing a Schedule Enable bit, software reads the USB 2.0 Status register to make sure that the corresponding Schedule Status bit has been set.

3. When system software determines that it should pause the EHC schedule, one or both of the Schedule Enable bits are written to 0. When this happens, the EHC responds as follows:
   a. The schedule disables are handled independently. In other words, the asynchronous and periodic disables may take effect in any order and vary greatly in latency.
   b. If the Periodic Schedule Enable is cleared, up to two more periodic transactions may be seen on the USB ports. Reads associated with the periodic schedule cease when the first fetch for a new transaction would normally be initiated; any reads required to execute an already partially-fetched transaction will continue to be generated. Writes associated with the periodic schedule may continue until all pending transactions in the periodic DMA engine's transaction queue are completed. The Periodic Schedule Status bit is cleared when the memory reads have completed and the memory writes have been internally posted.

Note: Multiple high-bandwidth packets are considered one transaction.

   c. If the Asynchronous Schedule Enable is cleared, up to two more asynchronous transactions may be seen on the USB ports. Reads associated with the asynchronous schedule cease when the first fetch for a new transaction would normally be initiated; any reads required to execute an already partially-fetched transaction will continue to be generated. Writes associated with the asynchronous schedule will continue until all pending transactions in the asynchronous DMA engine's transaction queue are completed. The Asynchronous Schedule Status bit is cleared when the memory reads have completed and the memory writes have been internally posted.
4. Before setting a Schedule Enable bit, software reads the USB 2.0 Status register to make sure that the corresponding Schedule Status bit is cleared.
5. When system software determines that it should reenable the EHC, one or both of the Schedule Enable are written to 1. When this happens, the EHC responds as described in the initial start-up case above.

The CMI does *not* implement a similar pause mechanism in the classic host controllers, which conflicts with the recommendation in the *EHCI Specification*.

26.10.2 Suspend Feature

The *EHCI Specification* describes the details of Port Suspend and Resume.

26.10.3 ACPI Device States

The USB 2.0 function only supports the D0 and D3 PCI Power Management states. Notes regarding implementation of the Device States:

1. The EHC hardware does not inherently consume any more power when it is in the D0 state than it does in the D3 state. However, software is required to suspend or disable all ports prior to entering the D3 state such that the maximum power consumption is reduced.
2. In the D0 state, all implemented EHC features are enabled.
3. In the D3 state, accesses to the EHC memory-mapped I/O range will master abort. Since the Debug Port uses the same memory range, the Debug Port is only operational when the EHC is in the D0 state.
4. In the D3 state, the EHC interrupt must never assert for any reason. The internal PME# signal is used to signal wake events, etc.
5. When the Device PowerState field is written to D0 from D3, an internal reset is generated. See Section 26.4.3 for general rules on the effects of this reset.

6. Attempts to write any other value into the Device PowerState field other than 00b (D0 state) and 11b (D3 state) will complete normally without changing the current value in this field.

Software performs the following sequence to put the EHC in the D3 state:

1. Software selectively suspends any enabled EHC ports.
2. Software reads back the selectively suspended ports to make sure the EHC has completed the suspend request.
3. Software clears the Run bit.
4. Software reads back the USB 2.0 Status Register to verify that the EHC is halted.
5. Software reads and saves the contents of the PCI configuration registers for restoring the context after transitioning back to the D0 state.
6. Software writes the Device PowerState field to the D3 state.

26.10.4 ACPI System States

The EHC behavior as it relates to other power management states in the system is summarized in the following list:

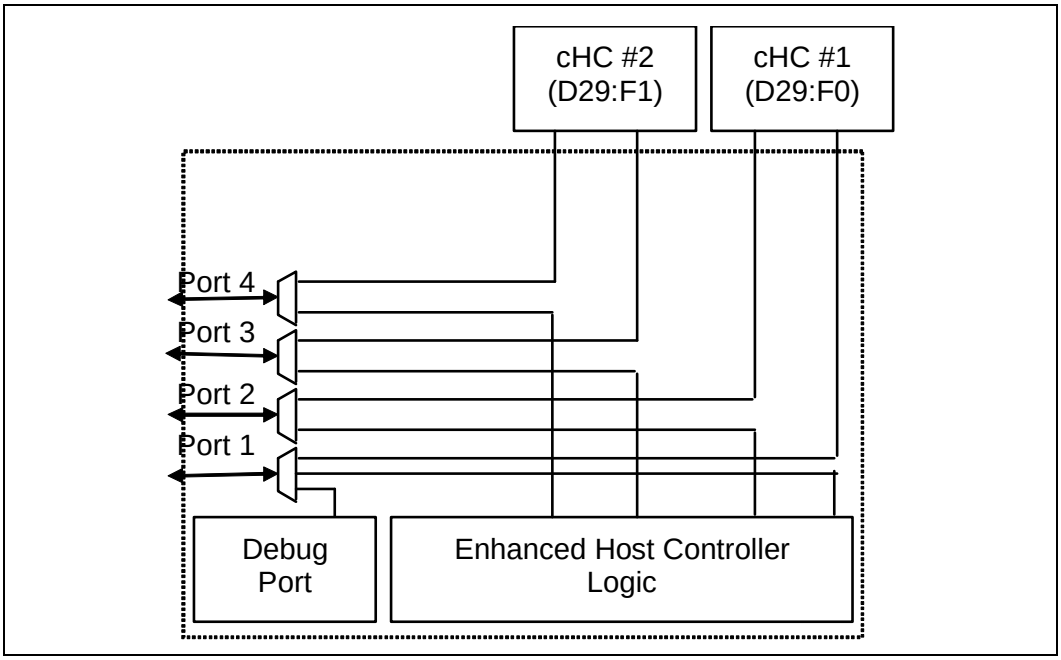
- The System is always in the S0 state when the EHC is in the D0 state. However, when the EHC is in the D3 state, the system may be in any power management state (including S0).
- When in D0, the Pause feature (described above) enables dynamic CPU low-power states to be entered.
- The PLL in the EHC is disabled when entering the S3-Cold/S4/S5 states (core power turns off).
- All core-well logic is reset in the S3/S4/S5 states.

26.11 Interaction with Classic Host Controllers

The Enhanced Host Controller shares the two USB ports with two UHCI Classic Host Controllers (cHCs). The cHC at Device 29: Function 0 shares ports 1 and 2 with the EHC. There is very little interaction between the Enhanced and Classic controllers other than the muxing control that is provided as part of the EHC.

Figure 26-1 depicts the USB Port Connections at a conceptual level. The dashed rectangle indicates all of the logic that is part of the Enhanced Host Controller cluster.

Figure 26-1. USB Port Connections



26.11.1 Port-Routing Logic

Integrated into the EHC functionality is "port-routing logic," which performs the muxing between the classic and enhanced Host Controllers. CMI conceptually implements this logic as described in the *EHCI Specification, Rev. 1.0.* If a device is connected that is not capable of USB 2.0's high-speed signaling protocol or if the EHCI software drivers are not present as indicated by the Configured Flag, then the cHC owns the port. Owning the port means that the differential output is driven by the owner and the input stream is only visible to the owner. The HC that is not the owner of the port internally sees a disconnected port.

Note: The port-routing logic is the only block of logic that observes the physical (real) connect/disconnect information. The port status logic inside each of the host controllers observes the electrical (artificial) connect/disconnect information that is generated by the port-routing logic.

Only the differential signal pairs are muxed/demuxed between the classic and enhanced host controllers. The other USB functional signals are handled as follows:

- The Overcurrent inputs (OC#[3:0]) are directly routed to both controllers. An overcurrent event is recorded in both controllers' status registers.
- The Port Routing logic is implemented in the Suspend power well so that reenumeration and remapping of the USB ports is not required following entering and exiting a system sleep state in which the core power is turned off.
- CMI also allows the USB Debug Port traffic to be routed in and out of Port #0. When in this mode, the Enhanced Host Controller is the owner of Port #0.

26.11.2 Device Connects

The *EHCI Specification, Rev. 1.0* describes the details of handling Device Connects. There are four general scenarios that are summarized below.

1. Configure Flag = 0 and a Classic-only Device is connected

   In this case, the classic Host Controller is the owner of the port both before and after the connect occurs; the EHC (except for the port-routing logic) never sees the connect occur. The UHCI driver handles the connection and initialization process.

2. Configure Flag = 0 and an Enhanced-capable Device is connected

   In this case, the classic Host Controller is the owner of the port both before and after the connect occurs; the EHC (except for the port-routing logic) never sees the connect occur. The UHCI driver handles the connection and initialization process. Since the classic Host Controller does not perform the high-speed chirp handshake, the device operates in compatible mode.

3. Configure Flag = 1 and a Classic-only Device is connected

   In this case, the enhanced Host Controller is the owner of the port before the connect occurs. The EHCI driver checks the Line Status bits to determine if a low-speed device is connected. If so, then the Port Owner bit is written to a 1 and the UHCI driver handles the reset sequence. If a low-speed device is not detected through the Line Status bits, the EHCI driver handles the connection and performs the port reset. After the reset process completes, the EHC hardware has cleared (not set) the Port Enable bit in the EHC's PORTSC register. The EHCI driver then writes a 1 to the Port Owner bit in the same register, causing the classic Host Controller to see a connect event and the EHC to see an "electrical" disconnect event. The UHCI driver and hardware handle the connection and initialization process from that point on. The EHCI driver and hardware handle the perceived disconnect.

4. Configure Flag = 1 and an Enhanced-capable Device is connected

   In this case, the enhanced Host Controller is the owner of the port before, and remains the owner after, the connect occurs. The EHCI driver handles the connection and performs the port reset. After the reset process completes, the EHC hardware has set the Port Enable bit in the EHC's PORTSC register. The port is functional at this point. The classic Host Controller continues to see an unconnected port.

26.11.3 Device Disconnects

The *EHCI Specification, Rev. 1.0* describes the details of handling Device Connects. There are three general scenarios that are summarized below.

1. Configure Flag = 0 and the device is disconnected

   In this case, the classic Host Controller is the owner of the port both before and after the disconnect occurs; the EHC (except for the port-routing logic) never sees a device attached. The UHCI driver handles disconnection process.

2. Configure Flag = 1 and a Classic Device is disconnected

   In this case, the classic Host Controller is the owner of the port before the disconnect occurs. The disconnect is reported by the classic Host Controller and serviced by the associated UHCI driver. The port-routing logic in the EHC cluster forces the Port Owner bit to 0, indicating that the EHC owns the unconnected port.

3. Configure Flag = 1 and an Enhanced Device is disconnected

   In this case, the enhanced Host Controller is the owner of the port before, and remains the owner after, the disconnect occurs. The EHCI hardware and driver handle the disconnection process. The classic Host Controller never sees a device attached.

26.11.4 Effect of Resets on Port-Routing Logic

As mentioned above, the Port Routing logic is implemented in the Suspend power well so that reenumeration and remapping of the USB ports is not required following entering and exiting a system sleep state in which the core power is turned off.

Table 26-54. Effect of Resets on Port-Routing Logic

Reset Event	Effect on Configure Flag	Effect on Port Owner Bits
Suspend Well Reset	cleared (0)	set (1)
Core Well Reset	no effect	no effect
D3-to-D0 Reset	no effect	no effect
HCRESET	cleared (0)	set (1)

26.12 USB 2.0 Legacy Keyboard Operation

CMI must support the possibility of a keyboard downstream from either a USB1 (low-speed or full-speed) or a USB 2.0 (high-speed) port. See Section 25.12, "USB Legacy Keyboard Operation" for the description of the legacy keyboard support.

The EHC provides the basic ability to generate SMIs on an interrupt event, along with more sophisticated control of the generation of SMIs, as documented in Section 26.2.1.27, "Offset 68h: ULSEC - USB 2.0 Legacy Support Extended Capability Register".

26.13 USB 2.0 Based Debug Port

CMI supports the elimination of the legacy COM ports by providing the ability for new debugger software to interact with devices on a USB 2.0 port.

High-level restrictions and features:

- Must be operational before USB 2.0 drivers are loaded.
  - — Must work even when the port is disabled.
  - — Must work even though non-configured port is default-routed to the classic controller.

Note: The Debug Port cannot be used to debug an issue that requires a classic USB device on Port #0 using the UHCI drivers.

- Must allow normal system USB 2.0 traffic in a system that may only have one USB port.
- Debug Port device (DPD) must be High-Speed capable and connect to a High-Speed port on CMI systems.
- Debug Port FIFO must always make forward progress (a bad status on USB is simply presented back to software).

The Debug Port FIFO is only given one USB access per microframe.

26.13.1 USB 2.0 Based Debug Port Overview

The Debug port facilitates OS and device driver debug. It allows the software to communicate with an external console using a USB 2.0 connection. Because the interface to this link does not go through the normal USB 2.0 stack, it allows communication with the external console during cases where the OS is not loaded, the USB 2.0 software is broken, or where the USB 2.0 software is being debugged.

Specific features of this implementation of a debug port are:

- Only works with an external USB 2.0 debug device (console)

- Implemented for a specific port on the host controller
- Operational anytime the port is not suspended AND the host controller is in D0 power state.
- Capability is interrupted when port is driving USB RESET

26.13.2 Debug Port Register Details

The Debug port's registers are located in the same memory range as the standard EHCI registers, which are defined by the Base Address Register (BAR). The base offset for these registers (A0h) is declared in the Debug Port Base Offset Capability Register at Configuration offset 5Ah. The specific EHCI port that supports this debug capability is indicated by a four-bit field (bits 20-23) in the HCSPARAMS register of an EHCI controller.

Behavioral Rules:

1. All of these registers are implemented in the core well and reset by EHC HCRESET, EHC D3-to-D0 state transition, and PLTRST#.
2. The hardware associated with this register provides no checks to ensure that software programs the interface correctly. How the hardware behaves when programmed illegally is undefined.

The default values are defined with an h for hex, a b for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

26.13.2.1 Offset A0h: CNTL_STS - Control/Status Register

Software must do Read-Modify-Write operations to this register to preserve the contents of bits not being modified. This include Reserved bits.

In order to preserve the usage of RESERVED bits in the future, software must always write the same value read from the bit until it is defined. Reserved bits will always return 0 when read.

Table 26-55. Offset A0h: CNTL_STS - Control/Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0h	
30	OWNER_CNT	0 = Ownership of the debug port is NOT forced to the EHCI controller (Default) 1 = Ownership of the debug port is forced to the EHCI controller (i.e., immediately taken away from the companion Classic USB Host controller). If the port was already owned by the EHCI controller, then setting this bit has no effect. This bit overrides all of the ownership-related bits in the standard EHCI registers. The value in this bit does not affect the value reported in the PORTSC Port Owner bit.		0h	RW
29	Reserved	Reserved		0h	
28	ENABLED_CNT	0 = Software can clear this by writing a 0 to it. The hardware clears this bit for the same conditions where the Port Enable/Disable Change bit (in the PORTSC register) is set. (Default) 1 = Debug port is enabled for operation. Software can directly set this bit if the port is already enabled in the associated PORTSC register (this is enforced by the hardware).		0h	RW
27	Reserved	Reserved		0h	
26 :17	Reserved	Reserved		0h	
16	DONE_STS	0 = Request Not complete. 1 = Set by hardware to indicate that the request is complete. Writing a 1 to this bit will clear it if it is set. Writing a 0 to this bit has no effect. Reset default = 0.		0h	RWC
15 :12	LINK_ID_STS	This field identifies the link interface. It is hardwired to 0h to indicate that it is a USB Debug Port.		0h	RO
11	Reserved	Reserved.		0h	
10	IN_USE_CNT	Set by software to indicate that the port is in use. Cleared by software to indicate that the port is free and may be used by other software. This bit is cleared after reset. (This bit has no effect on hardware.)		0h	RW
09 :07	EXCEPTION_STS	This field indicates the exception when the ERROR_GOOD#_STS bit is set. This field must be ignored if the ERROR_GOOD#_STS bit is 0. 000 No Error. ***Note:*** This must not be seen, since this field must only be checked if there is an error. 001 Transaction error: indicates the USB 2.0 transaction had an error (CRC, bad PID, timeout, etc.) 010 Hardware error. Request was attempted (or in progress) when port was suspended or reset. All others are reserved. Reset default = 000b		000b	RO

Table 26-55. Offset A0h: CNTL_STS - Control/Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** A0h **Offset End:** A3h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	ERROR_GOOD_N_STS	0 = The hardware clears this bit to 0 upon the proper completion of a read or write. 1 = The hardware sets this bit to indicate that an error has occurred. Details on the nature of the error are provided in the Exception field. Reset default = 0.		0h	RO
05	GO_CNT	Software sets this bit to cause the hardware to perform a read or write request. Writing a 0 to this bit has no effect. Writing a 1 to this bit when it is already set may result in undefined behavior. When set, the hardware clears this bit when the hardware sets the DONE_STS bit. Reset default = 0.		0h	RW
04	WRITE_READ_N_CNT	Software sets this bit to indicate that the current request is a write. Software clears this bit to indicate that the current request is a read. Reset default = 0.		0h	RW
03 :00	DATA_LEN_CNT	This field is used to indicate the size of the data to be transferred. For write operations, this field is set by software to indicate to the hardware how many bytes of data in Data Buffer are to be transferred to the console. A value of 0h indicates that a zero-length packet must be sent. A value of 1-8 indicates 1-8 bytes are to be transferred. Values 9-Fh are illegal and how hardware behaves if used is undefined. For read operations, this field is set by hardware to indicate to software how many bytes in Data Buffer are valid in response to a read operation. A value of 0h indicates that a zero length packet was returned and the state of Data Buffer is not defined. A value of 1-8 indicates 1-8 bytes were received. Hardware is not allowed to return values 9-Fh. The transferring of data always starts with byte 0 in the data area and moves toward byte 7 until the transfer size is reached. Reset default = 0h.		0h	RW

26.13.2.2 Offset A4h: USBPID - USB PIDs Register

This Dword register is used to communicate PID information between the USB debug driver and the USB debug port. The debug port uses some of these fields to generate USB packets, and uses other fields to return PID information to the USB debug driver.

Table 26-56. Offset A4h: USBPID - USB PIDs Register

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** A4h **Offset End:** A4h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	Reserved	Reserved.		0h	
23 :16	RECEIVED_PID_STS	The hardware updates this field with the received PID for transactions in either direction. When the controller is writing data, this field is updated with the handshake PID that is received from the device. When the host controller is reading data, this field is updated with the data packet PID (if the device sent data), or the handshake PID (if the device NAKs the request). This field is valid when the hardware clears the GO_DONE#_CNT bit.		0h	RO
15 :08	SEND_PID_CNT	The hardware sends this PID to begin the data packet when sending data to USB (i.e., WRITE_READ#_CNT is asserted). Software will typically set this field to either DATA0 or DATA1 PID values.		0h	RW
07 :00	TOKEN_PID_CNT	The hardware sends this PID as the Token PID for each USB transaction. Software will typically set this field to either IN, OUT or SETUP PID values.		0h	RW

26.13.2.3 Offset A8h: DATABUF - Data Buffer Bytes 7:0

Note: This register can be accessed as eight separate 8-bit registers or two separate 32-bit registers.

Table 26-57. Offset A8h: DATABUF - Data Buffer Bytes 7:0

Description:					
View: PCI	**BAR:** MBAR	**Bus:Device:Function:** 0:29:7		**Offset Start:** A8h **Offset End:** AFh	
Size: 64 bit	**Default:** 0000000000000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :00	DATABUFFER	These are the 8 bytes of the data buffer. Bits 7:0 correspond to least significant byte (byte 0). Bits 63:56 correspond to the most significant byte (byte 7). The bytes in the Data Buffer must be written with data before software initiates a write request. For a read request, the Data Buffer contains valid data when DONE_STS is set by the hardware, ERROR_GOOD#_STS is cleared by the hardware, and the DATA_LENGTH_CNT field indicates the number of bytes that are valid.		0h	RW

26.13.2.4 Offset B0h: CONFIG - Configuration Register

Table 26-58. Offset B0h: CONFIG - Configuration Register

Description:					
View: PCI	**BAR:** MBAR		**Bus:Device:Function:** 0:29:7	**Offset Start:** B0h **Offset End:** B0h	
Size: 32 bit	**Default:** 00007F01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :15	Reserved	Reserved		0h	
14 :08	USB_ADDRESS_CNF	7-bit field that identifies the USB device address used by the controller for all Token PID generation. This is a RW field that is set to 7Fh after reset.		7Fh	RW
07 :04	Reserved	Reserved		0h	
03 :00	USB_ENDPOINT_CNF	This 4-bit field identifies the endpoint used by the controller for all Token PID generation. This is a RW field that is set to 01h after reset.		1h	RW

26.13.3 USB 2.0 Based Debug Port Theory of Operation

There are two operational modes for the USB debug port:

1. Mode 1 is when the Enhanced USB Host Controller is in a disabled state from the viewpoint of a standard EHCI driver (i.e., Host Controller's *Run/Stop* bit is 0). In Mode 1, the Debug Port controller is required to generate 'keepalive' packets less than 2 milliseconds apart to keep the attached debug device from suspending. The keepalive packet must be a standalone 32-bit SYNC field.
2. Mode 2 is when the host controller is running (i.e., Host controller's *Run/Stop#* bit is 1). In Mode 2, the normal transmission of SOF packets (or SYNC keepalives if the port is functionally disabled) will keep the debug device from suspending.

26.13.3.1 Behavioral Rules

3. In both modes 1 and 2, the Debug Port controller must check for software requested debug transactions at least every 125 microseconds. If the debug port is enabled by the debug driver, and the standard host controller driver resets the USB port, USB debug transactions are held off for the duration of the reset and until after the first SOF is sent.
4. If the standard host controller driver suspends the USB port, then USB debug transactions are held off for the duration of the suspend/resume sequence and until after the first SOF is sent.
5. The ENABLED_CNT bit in the debug register space is independent of the similar port control bit in the associated Port Status and Control register.

Table 26-59 shows the debug port behavior related to the state of bits in the debug registers as well as bits in the associated Port Status and Control register.

Table 26-59. Debug Port Behavior

Debug bits		EHCI bits			Debug port behavior
OWNER_CNT	ENABLED_CNT	Port Enable	Run/ Stop#	Suspend	
0	X	X	X	X	Debug port is not being used. Normal operation.
1	0	X	X	X	Debug port is not being used. Normal operation.
1	1	0	0	X	Debug port in Mode 1. SYNC keepalives sent plus debug traffic
1	1	0	1	X	Debug port in Mode 2. SYNC keepalives or SOF packets may be sent plus debug traffic. CMI generates SYNC keepalives, not SOF packets. No other normal traffic is sent out this port, because the port is not enabled.
1	1	1	0	0	Illegal. Host controller driver must never put the controller into this state (enabled, not running and not suspended).
1	1	1	0	1	Port is suspended. No debug traffic sent.
1	1	1	1	0	Debug port in Mode 2. Debug traffic is interspersed with normal traffic.
1	1	1	1	1	Port is suspended. No debug traffic sent.

26.13.3.2 OUT Transactions

An OUT Transaction sends data to the debug device. It can occur only when the following are true:

- The debug port is enabled
- The debug software sets the GO_CNT bit
- The WRITE_READ#_CNT bit is set

The sequence of the transaction is:

1. Software sets the appropriate values in the following bits:

   USB_ADDRESS_CNF
   USB_ENDPOINT_CNF
   DATA_BUFFER[63:0]
   TOKEN_PID_CNT[7:0]
   SEND_PID_CNT[15:8]
   DATA_LEN_CNT
   WRITE_READ#_CNT(note: this will always be 1 for OUT transactions)
   GO_CNT (Note: this will always be 1 to initiate the transaction)

2. The debug port controller sends a token packet consisting of:

   A. SYNC
   B. TOKEN_PID_CNT field
   C. USB_ADDRESS_CNT field
   D. USB_ENDPOINT_CNT field
   E. 5-bit CRC field

3. After sending the token packet, the debug port controller sends a data packet consisting of:

   F. SYNC

G. SEND_PID_CNT field
H. The number of data bytes indicated in DATA_LEN_CNT from the DATA_BUFFER
I. 16-bit CRC

Note: A DATA_LEN_CNT value of zero is valid in which case no data bytes would be included in the packet.

4. After sending the data packet, the controller waits for a handshake response from the debug device.

- If a handshake is received, the debug port controller:

  J. Places the received PID in the RECEIVED_PID_STS field
  K. Resets the ERROR_GOOD#_STS bit
  L. Sets the DONE_STS bit

- If no handshake PID is received, the debug port controller:

  J. Sets the EXCEPTION_STS field to 001b
  K. Sets the ERROR_GOOD#_STS bit
  L. Sets the DONE_STS bit

26.13.3.3 IN Transactions

An IN transaction receives data from the debug device. It can occur only when the following are true:

- The debug port is enabled
- The debug software sets the GO_CNT bit
- The WRITE_READ#_CNT bit is reset

The sequence of the transaction is:

1. Software sets the appropriate values in the following bits:

   USB_ADDRESS_CNF
   USB_ENDPOINT_CNF
   TOKEN_PID_CNT[7:0]
   DATA_LEN_CNT
   WRITE_READ#_CNT(note: this will always be 0 for IN transactions)
   GO_CNT (note: this will always be 1 to initiate the transaction)
   The debug port controller sends a token packet consisting of:

   — SYNC
   — TOKEN_PID_CNT field
   — USB_ADDRESS_CNT field
   — USB_ENDPOINT_CNT field
   — 5-bit CRC field.

2. After sending the token packet, the debug port controller waits for a response from the debug device.

   If a response is received:
   The received PID is placed into the RECEIVED_PID_STS field

   — Any subsequent bytes are placed into the DATA_BUFFER
   — The DATA_LEN_CNT field is updated to show the number of bytes that were received after the PID.

3. If a valid packet was received from the device that was one byte in length (indicating it was a handshake packet), then the debug port controller:

— Resets the ERROR_GOOD#_STS bit

— Sets the DONE_STS bit

4. If a valid packet was received from the device that was more than one byte in length (indicating it was a data packet), then the debug port controller:
   — Transmits an ACK handshake packet
   — Resets the ERROR_GOOD#_STS bit
   — Sets the DONE_STS bit
5. If no valid packet is received, then the debug port controller:
   — Sets the EXCEPTION_STS field to 001b
   — Sets the ERROR_GOOD#_STS bit,
   — Sets the DONE_STS bit.

26.13.3.4 Debug Software

26.13.3.4.1 Enabling the Debug Port

There are two mutually exclusive conditions that debug software must address as part of its startup processing:

- The EHCI has been initialized by system software.
- The EHCI has not been initialized by system software.

Debug software can determine the current 'initialized' state of the EHCI by examining the *Configure Flag* in the EHCI USB 2.0 Command Register. If this flag is set, then system software has initialized the EHCI. Otherwise, the EHCI must not be considered initialized. Debug software will initialize the debug port registers depending on the state of the EHCI. However, before this can be accomplished, debug software must determine which root USB port is designated as the debug port.

26.13.3.4.2 Determining the Debug Port

Debug software can easily determine which USB root port has been designated as the debug port by examining bits 20:23 of the EHCI Host Controller Structural Parameters register. This 4-bit field represents the numeric value assigned to the debug port (i.e., 0000 == port 0, 0001 == port 1, 0010 == port 2 ..... 1111 == port 15). This value is 0000 (port 0).

26.13.3.4.3 Debug Software Startup with Non-Initialized EHCI

Debug software can attempt to use the debug port if, after setting the *OWNER_CNT* bit, the *Current Connect Status* bit in the appropriate (See Section 26.13.3.4.2, "Determining the Debug Port") PORTSC register is set. If the *Current Connect Status* bit is not set, then debug software may choose to terminate, or it may choose to wait until a device is connected.

If a device is connected to the port, then debug software must reset/enable the port. Debug software does this by setting and then clearing the *Port Reset* bit in the PORTSC register. Software must set the *Run/Stop* bit in the EHCI Command Register before clearing the *Port Reset* bit in order to complete the reset and to enable the port. To guarantee a successful reset, debug software must also keep the *Port Reset* bit set for at least 50 ms. Due to possible delays, this bit may not change to zero immediately; reset is complete when this bit reads as zero. Software must not continue until this bit reads zero.

If a high-speed device is attached, the EHCI will automatically set the *Port Enabled/Disabled* bit in the PORTSC register and the debug software can proceed. Debug software must set the *ENABLED_CNT* bit in the Debug Port Control/Status register, and then reset (clear) the *Port Enabled/Disabled* bit in the PORTSC register and the *Run/Stop* bit in the EHCI Command Register. The EHCI bits are cleared in order to present the proper default idle conditions to the EHCI driver as it loads.

26.13.3.4.4 Debug Software Startup with Initialized EHCI

Debug software can attempt to use the debug port if the *Current Connect Status* bit in the appropriate (See Section 26.13.3.4.2, "Determining the Debug Port") PORTSC register is set. If the *Current Connect Status* bit is not set, then debug software may terminate or it may wait until a device is connected.

If a device is connected, then debug software must set the *OWNER_CNT* bit and then the *ENABLED_CNT* bit in the Debug Port Control/Status register.

26.13.3.4.5 Determining Debug Peripheral Presence

After enabling the debug port functionality, debug software can determine if a debug peripheral is attached by attempting to send data to the debug peripheral. If all attempts result in an error (*Exception* bits in the Debug Port Control/Status register indicates a Transaction Error), then the attached device is not a debug peripheral. If the debug port peripheral is not present, then debug software may terminate or it may wait until a debug peripheral is connected.

§ §

27.0 Power Management

27.1 Features

- ACPI Power and Thermal Management Support.
  - — Processor THRMTRIP# emergency shutdown.
  - — ACPI 24-Bit Timer.
  - — Software initiated throttling of processor performance for Thermal and Power Reduction.
  - — SCI and SMI# Generation.
- PCI PME# Signal for Wake Up from Low-Power states.
- SYS_Reset# input to eliminate external glue logic.
- System Clock Control.
  - — ACPI C0 state — Full On: Processor operating. Individual devices may be shut to save power.
  - — ACPI C1 state — Auto-Halt: Processor has executed a AutoHalt instruction and is not executing code. The processor snoops the bus and maintains cache coherency.
  - — ACPI C2 state — Stop-Grant state (using STPCLK# signal) halts processor's instruction stream.
- System Sleeping State Control.
  - — ACPI S0 state – All power planes active (awake).
  - — ACPI S3 state – Suspend to RAM (STR).
  - — Supports S3-Cold state.
  - — ACPI S4 state – Suspend-to-Disk (STD).
  - — ACPI S5 state – Soft Off (SOFF).
  - — Power Failure Detection and Recovery.
- Streamlined Legacy Power Management Support for APM-Based Systems.

27.2 IMCH-IICH Messages

NSI messaging protocol is supported. Messages associated with power management and state transitions are summarized in Table 27-1. The Legacy Protocol messages are shown only for reference. NSI messages that are not associated with power management are not shown.

Table 27-1. IMCH-IICH Messages

Legacy Message	New Message	Direction	Description/Comment:
---	Reset-Warn	IICH→IMCH	Warning from the IICH to the IMCH that the IICH is about to assert the PLTRST# signal. The IMCH is expected to acknowledge this with the Reset-Warn-Ack. However, it the IMCH fails to do this within the timeout period, the IICH will assert the reset.
---	Reset-Warn-Ack	IMCH→IICH	Acknowledge from the IMCH that it has seen the Reset-Warn message is now ready for the IICH to cause the reset.
Stop-Grant	Stop-Grant (REQ-C2)	IMCH→IICH	If the processor is in C0 - indication that the processor has issued last Stop-Grant cycle. The IMCH may receive more than one Stop-Grant cycle from the processor(s). It is the IMCH's responsibility to only send the last Stop-Grant.
Go-C0	Go-C0	IICH→IMCH	Indication that system is going back to C0 state.
--	Ack-C0	IMCH→IICH	Acknowledge that IMCH observed the Go-C0 message and is ready to proceed.
--	Go-C2	IICH→IMCH	This is an indication that the processor has been put into Stop-Grant state. When coming from C0, this tells the IMCH that it is safe to assert SLP#.
--	Ack-C2	IMCH→IICH	IMCH indicates it observed the Go-C2 message and is now ready to proceed.f going toward C0, the IICH is free to deassert STPCLK#.
Go-C3	Go-S3	IICH→IMCH	Indication that the IICH is getting ready to put the system into S3, S4 or S5 state.
Ack-C3	Ack-S3	IMCH→IICH	Indication that the IMCH observed the Go-S3 message and is ready to proceed.
--	REQ-C0 (Break-Ind)	IMCH →IICH	This is an indication from the IMCH to the IICH that the IMCH thinks the processor must be brought to a C0 state. This would be sent for several cases: 1. If the IMCH had received a "Pending Break Event" indication from the processor. This is needed when PBE# is not muxed with FERR# and is instead muxed with some pin that goes only to the IMCH. 2. The IMCH has some internal device or link to external device that can cause a break event that is not associated with an interrupt. ***Note:*** The IMCH is not required to implement this message.

27.3 Power Management Register Details

This section shows the power management registers. The power management registers are distributed within the PCI Device 31: Function 0 space, as well as a separate I/O range. Each register is described below. Unless otherwise indicated, bits are in the main (core) power well.

Bits not explicitly defined in each register are assumed to be reserved.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers my return on-zero values are read only. Writes to reserved locations may cause system failure and unpredictable behavior.

Note: Reserved bits are read only.

27.3.1 Power Management PCI Configuration Registers

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183).

Table 27-2. Bus 0, Device 31, Function 0: Summary of LPC Interface Power Management PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
A0h	A0h	“Offset A0h: GEN_PMCON_1 - General PM Configuration 1 Register” on page 1048	0200h
A2h	A2h	“Offset A2h: GEN_PMCON_2 - General PM Configuration 2 Register” on page 1049	00h
A4h	A4h	“Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register” on page 1051	00h
B8h	BBh	“Offset B8h: GPI_ROUT - GPI Routing Control Register” on page 1053	00000000h

27.3.1.1 Offset A0h: GEN_PMCON_1 - General PM Configuration 1 Register

Table 27-3. Offset A0h: GEN_PMCON_1 - General PM Configuration 1 Register (Sheet 1 of

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A0h **Offset End:** A0h	
Size: 16 bit	**Default:**	0200h		**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	
10	BIOS_PCI_EXP_EN	This bit acts as a global enable for the SCI associated with the PCI Express* ports. 0 = The various PCI Express* ports and cannot cause the PCI_EXP_STS bit to go active. 1 = The various PCI Express* ports and can cause the PCI_EXP_STS bit to go active.		0h	RW
09	PWRBTN_LVL	This bit indicates the current state of the PWRBTN# signal. 0 = Low 1 = High		1	RO

Table 27-3. Offset A0h: GEN_PMCON_1 - General PM Configuration 1 Register (Sheet 2 of

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A0h **Offset End:** A0h	
Size: 16 bit	**Default:**	0200h		**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	Reserved	Reserved		0h	
07	Reserved	Reserved		0h	
06		1 = Reserved		0h	RW
05	CPUSLP_EN	CPU SLP# Enable. Software sets this bit to enable the CPUSLP# signal to go active in the S1 state. 0 = Disable. 1 = Enables the CPUSLP# signal to go active when the processor is placed in S1 state. The signal is not asserted when the EP80579 is in the S3, S4 or S5 state.		0h	
04	SMI_LOCK	When this bit is set, writes to the GLB_SMI_EN bit will have no effect. Once the SMI_LOCK bit is set, writes of 0 to SMI_LOCK bit will have no effect (i.e., once set, this bit can only be cleared by PLTRST#).		0h	RWO
03	RSVD	Reserved		0h	
02	RSVD	Reserved		0h	
01 : 00	PER_SMI_SEL	Software sets these bits to control the rate at which the periodic SMI# is generated: 00 = 64 seconds (default) 01 = 32 seconds 10 = 16 seconds 11 = 8 seconds Tolerance for the timer is ±1 second.		00h	RW

a. Bits 10, 07:00 – Core; Bit 9 – Resume

27.3.1.2 Offset A2h: GEN_PMCON_2 - General PM Configuration 2 Register

Table 27-4. Offset A2h: GEN_PMCON_2 - General PM Configuration 2 Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A2h **Offset End:** A2h	
Size: 8 bit	**Default:**	00h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	DIB	**DRAM Initialization bit:** This bit does not effect hardware functionality in any way.BIOS is expected to set this bit prior to starting the DRAM initialization sequence and to clear this bit after completing the DRAM initialization sequence. BIOS can detect that a DRAM initialization sequence was interrupted by a reset by reading this bit during the boot sequence. If the bit is 1, then the DRAM initialization was interrupted. See Section 27.5.1 for the expected BIOS response. This bit is reset by the assertion of the RSMRST# pin.		0h	RW
06 : 05	RSVD	Reserved		00h	
04	SRS	**System Reset Status:** 0 = SYS_RESET# button Not pressed. 1 = This bit is set when the SYS_RESET# button is pressed. BIOS is expected to read this bit and clear it if it is set. ***Note:*** This bit is also reset by RSMRST# and CF9h resets.		0h	RWC
03	CTS	**CPU Thermal Trip Status:** 0 = Software clears this bit by writing a 1 to it. 1 = This bit is set when PLTRST# is inactive and THRMTRIP# goes active while the system is in an S0 or S1 state. ***Note:*** This bit is also reset by RSMRST# and CF9h resets. It is not reset by the shutdown and reboot associated with the CPUTHRMTRIP# event.		0h	RWC

Table 27-4. Offset A2h: GEN_PMCON_2 - General PM Configuration 2 Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A2h **Offset End:** A2h	
Size: 8 bit	**Default:**	00h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	MAWVS	**Minimum SLP_S4# Assertion Width Violation Status**: 0 = Software clears this bit by writing a 1 to it. 1 = Hardware sets this bit when the SLP_S4# assertion width is less than the time programmed in the SLP_S4# Minimum Assertion Width field (D31.F0.A4h.5:4). When exiting G3, the timer begins when the RSMRST# input deasserts. ***Note:*** This bit is functional regardless of the value in the SLP_S4# Assertion Stretch Enable. This bit is reset by the assertion of the RSMRST# pin, but can be set in some cases before the default value is readable.		0h	RWC
01	CPUPWR_FLR	**CPU Power Failure:** 0 = Software (typically) BIOS clears this bit by writing a 0 to it. 1 = Indicates that the VRMPWRGD input signal (≠) from the processor's VRM went low. ***Note:*** VRMPWRGD is sampled using the RTC clock. Therefore, low times that are less than one RTC clock period may not be detected.		0h	RW
00	PWROK_FLR	**Power OK Failure:** 0 = Software clears this bit by writing a 1 to it, or when the system goes into a G3 state. 1 = This bit will be set any time PWROK goes low, when the system was in S0, or S1 state. The bit will be cleared only by software by writing a 1 to this bit or when the system goes to a G3 state. See Section 27.6.3 for more details about the PWROK pin functionality. ***Note:*** In the case of true PWROK failure, PWROK will go low first before the VRMPWRGD.		0h	RWC

27.3.1.3 Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register

Table 27-5. Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A4h **Offset End:** A4h	
Size: 8 bit	**Default:**	00h		**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	SWSMI_RATE_SEL	This 2-bit value indicates when the SWSMI timer will time out. Valid values are: • 00 1.5 ms +/- 0.6 ms • 01 16 ms +/- 4 ms • 10 32 ms +/- 4 ms • 11 64 ms +/- 4 ms These bits are not cleared by any type of reset except RTEST#.		00h	RW
05 : 04	SMAW	**SLP_S4# Minimum Assertion Width:** This 2-bit value indicates the minimum assertion width of the SLP_S4# signal to guarantee that the DRAMs have been safely power-cycled. This value may be modified per platform depending on DRAM types, power supply capacitance, etc. Valid values are: • 11 1 to 2 seconds • 10 2 to 3 seconds • 01 3 to 4 seconds • 00 4 to 5 seconds This value is used in two ways: 1. If the SLP_S4# assertion width is ever shorter than this time, a status bit (D31.F0.A2h.2) is set for BIOS to read when S0 is entered 2. If enabled by bit 3 in this register, the hardware will prevent the SLP_S4# signal from deasserting within this minimum time period after asserting ***Note:*** The logic that measures this time is in the suspend power well. Therefore, when leaving the G3 state, the minimum time is measured from the deassertion of RSMRST#. RTEST# forces this field to the conservative default state (00b).		00h	
03	SASE	**SLP_S4# Assertion Stretch Enable:** 0 = The SLP_S4# minimum assertion time is 1 to 2 RTCCLK. 1 = The SLP_S4# signal will minimally assert for the time specified in bits 5:4 of this register. This bit is cleared by RTEST#.		0h	RW
02	RPS	**RTC Power Status:** 0 = RTEST# OK 1 = RTEST# indicates a weak or missing battery. The bit remains set until the software clears it by writing a 0 back to this bit position. This bit is not cleared by any type of reset.		X	RW

Table 27-5. Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** A4h **Offset End:** A4h	
Size: 8 bit	**Default:**	00h		**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	PWR_FLR	**PWR_FLR:** 0 = Indicates that the trickle current has not failed since the last time the bit was cleared. 1 = Indicates that the trickle current (from the main battery or trickle supply) was removed or failed. Software writes a 1 to this bit to clear it. This bit is in the RTC well, and is not cleared by any type of reset except RTEST#. ***Notes:*** 1. RSMRST# is sampled using the RTC clock. Therefore, low times that are less than one RTC clock period may not be detected. 2. Clearing CMOS in CMI platforms can be done by using a jumper on RTEST# or GPI. Implementations must not attempt to clear CMOS by using a jumper to pull VccRTC low.		0h	RWC
00	AG3E	**AFTERG3_EN:** Determines what state to go to when power is reapplied after a power failure (G3 state). 0 = System will return to an S0 state (boot) after power is reapplied. 1 = System will return to the S5 state (except if it was in S4, in which case it will return to S4-like state). In addition to software writes, this bit is set by the following hardware conditions: • Power Button Override • SMBus Unconditional Powerdown Message • Catastrophic Temperature condition from an internal sensor • Assertion of CPU Thermal Trip input		0h	RW

27.3.1.4 Offset B8h: GPI_ROUT - GPI Routing Control Register

Table 27-6. Offset B8h: GPI_ROUT - GPI Routing Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:31:0	**Offset Start:** B8h **Offset End:** BBh	
Size: 32 bit	**Default:**	00000000h		**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	GPI	**GPI[15] through GPI[1]:** See bits 1:0 for description.		0h	RW
01 : 00	GPI0_Route	If the corresponding GPIO is implemented and is set to an input, a '1' in the GP_LVL bit can be routed to cause an SMI# or SCI. If the GPIO is not set to an input, this field has no effect. • 00 – No effect (or GPIO unimplemented) • 01 – SMI# (if corresponding ALT_GPI_SMI_EN bit also set) • 10 – SCI (if corresponding GPE0_EN bit also set) • 11 – Reserved If the system is in an S1,S3,S4 or S5 state and if the GPE0_EN bit is also set, then the GPI can cause a Wake event, even if the GPI is NOT routed to cause an SMI# or SCI. Exception: If the system is in S5 state due to a powerbutton override, then the GPIs will not cause wake events. ***Note:*** Core well GPIs are not capable of waking the system from sleep states where the core well is not powered.		00h	RW

27.3.2 APM Power Management I/O-Mapped Registers

Table 27-7 shows the I/O registers associated with APM support. This register space is enabled in the PCI Device 31: Function 0 space (APMC_EN), and cannot be moved (fixed I/O location).

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables" on page 183).

Table 27-7. Summary of APM Registers Mapped in I/O Space

Offset Start	**Offset End**	**Register ID - Description**	**Default Value**
B2h	B2h	"Offset B2h: APM_CNT - Advanced Power Management Control Port Register" on page 1054	00h
B3h	B3h	"Offset B3h: APM_STS - Advanced Power Management Status Port Register" on page 1054	00h

27.3.2.1 Offset B2h: APM_CNT - Advanced Power Management Control Port Register

Used to pass an APM command between the OS and the SMI handler. Writes to this port not only store data in the APMC register, but also generates an SMI# when the APMC_EN bit is set.

Table 27-8. Offset B2h: APM_CNT - Advanced Power Management Control Port Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** B2h **Offset End:** B2h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	APM_CNT	Used to pass an APM command between the OS and the SMI handler. Writes to this port not only store data in the APMC register, but also generates an SMI# when the APMC_EN bit is set.		00h	RW

27.3.2.2 Offset B3h: APM_STS - Advanced Power Management Status Port Register

Used to pass data between the OS and the SMI handler. Basically, this is a scratchpad register and is not effected by any other register or function (other than a PCI reset).

Table 27-9. Offset B3h: APM_STS - Advanced Power Management Status Port Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** B3h **Offset End:** B3h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	APM_STS	Used to pass data between the OS and the SMI handler. Basically, this is a scratchpad register and is not affected by any other register or function (other than a platform reset).		00h	RW

27.3.3 General Power Management I/O-Mapped Registers

Table 27-10 shows the registers associated with ACPI and Legacy power management support. These registers are enabled in the PCI Device 31: Function 0 space (ACPI Enable in Section 19.2.2), and can be moved to any I/O location (128-byte aligned) determined by ABASE in Section 19.2.2.1 (referenced in this chapter by PMBASE). The registers are defined to be compliant with the *Advanced Configuration and Power Interface (ACPI) Specification, Rev. 2.0*, and generally use the same bit names. All reserved bits and registers will always return 0 when read, and will have no effect when written.

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables" on page 183).

Table 27-10. Bus 0, Device 31, Function 0: Summary of LPC Interface Power Management General Configuration Registers Mapped Through PMBASE I/O BAR

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PM1_STS – Power Management 1 Status Register" on page 1056	0000h
02h	02h	"Offset 02h: PM1_EN - Power Management 1 Enables Register" on page 1058	0000h
04h	04h	"Offset 04h: PM1_CNT - Power Management 1 Control Register" on page 1059	0000h
08h	B8h	"Offset 08h: PM1_TMR - Power Management 1 Timer Register" on page 1060	00000000h
10h	10h	"Offset 10h: PROC_CNT - Processor Control Register" on page 1060	00000000h
14h	14h	"Offset 14h: LV2 - Level 2 Register" on page 1063	00h
28h	28h	"Offset 28h: GPE0_STS - General Purpose Event 0 Status Register" on page 1063	00000000h
2Ch	2Ch	"Offset 2Ch: PMBASE_GPE0_EN - General Purpose Event 0 Enables Register" on page 1067	00000000h
30h	30h	"Offset 30h: SMI_EN - SMI Control and Enable Register" on page 1068	00000000h
34h	34h	"Offset 34h: SMI_STS - SMI Status Register" on page 1071	00000000h
38h	38h	"Offset 38h: ALT_GPI_SMI_EN - Alternate GPI SMI Enable Register" on page 1073	0000h
3Ah	3Ah	"Offset 3Ah: ALT_GPI_SMI_STS - Alternate GPI SMI Status Register" on page 1074	0000h
44h	44h	"Offset 44h: DEVTRAP_STS - DEVTRAP_STS Register" on page 1074	0000h

27.3.3.1 Offset 00h: PM1_STS – Power Management 1 Status Register

Table 27-11. Offset 00h: PM1_STS – Power Management 1 Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 00h **Offset End:** 00h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	WAK_STS	0 = Software clears this bit by writing a 1 to it. 1 = This bit can only be set by hardware when the system is in one of the Sleep states (via the SLP_EN bit) and an enabled Wake event occurs. Upon setting this bit, CMI will transition the system to the ON state. This bit is not affected by hard resets caused by a CF9 write, but is reset by RSMRST#. If a power failure occurs (such as removed batteries) without the SLP_EN bit set, the WAK_STS bit will not be set when the power returns if the AFTER_G3 bit is 0. If the AFTER_G3 bit is 1, then the WAK_STS bit will be set after waking from a power failure. If necessary, the BIOS can clear the WAK_STS bit in this case.		0h	RWC
14	Reserved	Reserved.		0h	
13 : 12	Reserved	Reserved		00h	
11	PRBTNOR_STS	0 = Software clears this bit by writing a 1 to it. 1 = This bit is set any time a Power Button Override Event occurs (i.e., the power button is pressed for at least 4 consecutive seconds), or due to the corresponding bit in the SMBus slave message, or due to an internal thermal sensor catastrophic condition. These events cause an unconditional transition to the S5 state, as well as sets the AFTERG3 bit. The BIOS or SCI handler clears this bit by writing a 1 to it. This bit is not affected by hard resets via CF9h writes, and is not reset by RSMRST#. Thus, this bit is preserved through power failures.		0h	RWC
10	RTC_STS	0 = Software clears this bit by writing a 1 to it. 1 = Set when the RTC generates an alarm (assertion of the IRQ8# signal), and is not affected by any other enable bit. See RTC_EN for the effect when RTC_STS goes active. This bit is only set by hardware and can only be reset by writing a one to this bit position. This bit is not affected by hard resets caused by a CF9 write, but is reset by RSMRST#. Additionally if the RTC_EN bit (PMBASE + 02h, bit 10) is set, the setting of the RTC_STS bit will generate a wake event.		0h	RWC
09	Reserved	Reserved		0h	

Table 27-11. Offset 00h: PM1_STS – Power Management 1 Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 00h **Offset End:** 00h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	PWRBTN_STS	This bit is set when the PWRBTN# signal is asserted (low), independent of any other enable bit. See PWRBTN_EN for the effect when PWRBTN_STS goes active. PWRBTN_STS is always a wake event. This bit is only set by hardware and can be cleared by software writing a one to this bit position. This bit is not affected by hard resets caused by a CF9 write, but is reset by RSMRST#. If the PWRBTN# signal is held low for more than 4 seconds, CMI clears the PWRBTN_STS bit, sets the PWRBTNOR_STS bit, the system transitions to the S5 state, and only PWRBTN# is enabled as a wake event. If PWRBTN_STS bit is cleared by software while the PWRBTN# pin is still held low, this will not cause the PWRBTN_STS bit to be set. The PWRBTN# signal must go inactive and active again to set the PWRBTN_STS bit. ***Note:*** The SMBus Unconditional Power down message, the CPU Thermal Trip and the Internal Thermal Sensors' Catastrophic Condition result in behavior matching the Powerbutton Override, which includes clearing this bit.		0h	RWC
07 : 06	Reserved	Reserved		0h	
05	GBL_STS	0 = The SCI handler must then clear this bit by writing a 1 to the bit location. 1 = Set when an SCI is generated due to BIOS wanting the attention of the SCI handler. BIOS has a corresponding bit, BIOS_RLS, which will cause an SCI and set this bit. This bit will not cause wake events or SMI#. This bit is not effected by SCI_EN. ***Note:*** GBL_STS being set will cause an SCI, even if the SCI_EN bit is not set. Software must take great care not to set the BIOS_RLS bit (which causes GBL_STS to be set) if the SCI handler is not in place.		0h	RWC
04 01	Reserved	Reserved		000h	
00	TMROF_STS	0 = The SCI or SMI# handler clears this bit by writing a 1 to the bit location. 1 = This bit gets set any time bit 22 of the 24-bit timer goes low (bits are numbered from 0 to 23). This will occur every 2.3435 seconds. Hence, it is highly likely that a read to this register after reset will yield a 1 in this field. When the TMROF_EN bit (PMBASE + 02h, bit 0) is set, then the setting of the TMROF_STS bit will additionally generate an SCI or SMI# (depending on the SCI_EN).		0h	RWC

a. Bits 0-7: Core, Bits 8-15: Resume (except 11 in RTC)

27.3.3.2 Offset 02h: PM1_EN - Power Management 1 Enables Register

Table 27-12. Offset 02h: PM1_EN - Power Management 1 Enables Register

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 02h **Offset End:** 02h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	Reserved	Reserved		0h	
14	Reserved	Reserved		0h	
13 11	Reserved	Reserved		0h	
10	RTC_EN	This is the RTC alarm enable bit. It works in conjunction with the SCI_EN bit: RTC_EN / SCI_EN / Effect when RTC_STS is set 0 / x / No SMI# or SCI. If system was in S1,S3,S4 or S5, no wake event occurs. 1 / 0 / SMI#. If system was in S1,S3,S4 or S5, then a wake event occurs before the SMI#. 1 / 1 / SCI. If system was in S1, S3, S4 or S5, then a wake event occurs before the SMI#.		0h	RW
09	Reserved	Reserved		0h	
08	PWRBTN_EN	This bit is the power button enable. It works in conjunction with the SCI_EN bit: PWRBTN_EN / SCI_EN / Effect when PWRBTN_STS is set 0 / x / No SMI# or SCI 1 / 0 / SMI# 1 / 1 / SCI ***Note:*** PWRBTN_EN has no effect on the PWRBTN_STS bit being set by the assertion of the power button. The Power Button is always enabled as a Wake event.		0h	RW
07 : 06	Reserved	Reserved		0h	
05	GBL_EN	The global enable bit. When both the GBL_EN and the GBL_STS are set, an SCI is generated. 0 = Disable. 1 = Enable SCI on GBL_STS going active.		0h	RW
04 : 01	Reserved	Reserved		0h	
00	TMROF_EN	This is the timer overflow interrupt enable bit. It works in conjunction with the SCI_EN bit: TMROF_EN / SCI_EN / Effect when TMROF_STS is set 0 / x / No SMI# or SCI. If system was in S1, S3, S4 or S5, no wake event. 1 / 0 / SMI#. If system was in S1,S3, S4,S5, then a wake event occurs before the SMI#. 1 / 1 / SCI. If system was in S1,S3,S4 or S5, then a wake event occurs before the SMI#.		0h	RW

a. Bits 0-7: Core, Bits 8-15: Resume

27.3.3.3 Offset 04h: PM1_CNT - Power Management 1 Control Register

Table 27-13. Offset 04h: PM1_CNT - Power Management 1 Control Register

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 04h **Offset End:** 04h	
Size: 32 bit	**Default:** 0000h			**Power Well:** Core[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 14	Reserved	Reserved		0h	
13	SLP_EN	This is a write-only bit and reads to it always return a zero. Setting this bit causes the system to sequence into the Sleep state defined by the SLP_TYP field.		0h	WO
12 : 10	SLP_TYP	This 3-bit field defines the type of Sleep the system must enter when the SLP_EN bit is set to 1. These bits are reset by RTEST# only. Bits / Mode / Typical Mapping 000 — ON — S0 001 — Just assert STPCLK#. Puts processor in Stop-Grant state. Can also assert CPUSLP#, to put processor in Sleep state. — S1 010 — Reserved 011 — Reserved 100 — Reserved 101 — Suspend-To-RAM — S3 110 — Suspend-To-Disk — S4 111 — Soft Off — S5		0h	RW
09 : 03	Reserved	Reserved		0h	
02	GBL_RLS	This bit is used by the ACPI software to raise an event to the BIOS software. BIOS software has a corresponding enable and status bits to control its ability to receive ACPI events. This bit always reads as 0.		0h	WO
	Reserved	Reserved		0h	
00	SCI_EN	Selects the SCI interrupt or the SMI# for various events. 0 = These events generate an SMI#. 1 = The events generate an SCI interrupt.		0h	RW

a. Bits 0-7: Core, Bits 8-12: RTC, Bits 13-15: Resume

27.3.3.4 Offset 08h: PM1_TMR - Power Management 1 Timer Register

Table 27-14. Offset 08h: PM1_TMR - Power Management 1 Timer Register

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 08h **Offset End:** B8h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved. Will always read as 0.		00h	
23 : 00	TMR_VAL	This read-only field returns the running count of the PM timer. This counter runs off a 3.579545 MHz clock (derived from 14.31818 MHz divided by 4). It is reset (to 0) during a PCI reset, and then continues counting as long as the system is in the S0 state. Hence, it is highly likely that a read to this register after reset will yield a non-zero value. After an S1 state, the counter will not be reset (it will continue counting from the last value in S0 state). Anytime bit 22 of the timer goes HIGH to LOW (bits referenced from 0 to 23), the TMROF_STS bit is set. The High-to-Low transition will occur every 2.3435 seconds. Writes to this register have no effect.		00h	RO

27.3.3.5 Offset 10h: PROC_CNT - Processor Control Register

Table 27-15. Offset 10h: PROC_CNT - Processor Control Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 10h **Offset End:** 10h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 18	Reserved	Reserved		00h	
17	THTL_STS	0 = No clock throttling is occurring (maximum processor performance). 1 = Indicates that the clock state machine is throttling the CPU performance. This could be due to the THT_EN bit or the FORCE_THTL bit being set.		0h	RO
16 : 09	Reserved	Reserved		00h	
08	FORCE_THTL	Software can set this bit to 1 to force the throttling. 0 = The throttling (at a duty cycle specified in PROCHOT_DTY) does not start immediately and does generate an SMI#. 1 = The throttling (at a duty cycle specified in PROCHOT_DTY) starts immediately and does not generate an SMI#.		0h	RW

Table 27-15. Offset 10h: PROC_CNT - Processor Control Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 10h **Offset End:** 10h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	PROCHOT_DTY	This write-once 3-bit field determines the duty cycle of the throttling when the FORCE_THTL bit is set. The duty cycle indicates the approximate percentage of time the STPCLK# signal is asserted while in the thermal throttle mode. The STPCLK# throttle period is 1024 PCICLKs. Throttling only occurs if the system is in the C0 state. If in the C2 state, no throttling occurs. Once the PROCHOT_DTY field is written, subsequent writes have no effect until PLTRST# goes active. **PROCHOT_DTY Bits[2:0]** / **Throttle Mode** / **PCI Clocks** 000 / Default (will be 50%) / 512 001 / 87.5% / 896 010 / 75.0% / 768 011 / 62.5% / 640 100 / 50% / 512 101 / 37.5% / 384 110 / 25% / 256 111 / 12.5% / 128		000h	RW

Table 27-15. Offset 10h: PROC_CNT - Processor Control Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 10h **Offset End:** 10h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	THT_EN	When this bit is set and the system is in a C0 state, it enables a software controlled STPCLK# throttling. The duty cycle is selected in the THTL_DTY field. 0 = Disable 1 = Enable		0h	RW
03 : 01	THTL_DTY	This 3 –bit field determines the duty cycle of the throttling when the THT_EN bit is set. The duty cycle indicates the approximate percentage of time the STPCLK# signal is **asserted (low)** while in the throttle mode. The STPCLK# throttle period is 1024 PCICLKs. **PROCHOT_DTY Bits[3:0]** / **Throttle Mode** / **PCI Clocks (STPCLK# low)** 000 / Default (will be 50%) / 512 001 / 87.75% / 896 010 / 75% / 768 011 / 62.5% / 640 100 / 50% / 512 101 / 37.5% / 384 110 / 25% / 256 111 / 12.5% / 128		000h	RW
00	Reserved	Reserved		0h	

27.3.3.6 Offset 14h: LV2 - Level 2 Register

Reads to this register return all zeros, writes to this register have no effect. Reads to this register generate a “enter a level 2 power state” (C2) to the clock control logic. This will cause the STPCLK# signal to go active, and stay active until a break event occurs. Throttling (due to THTL_EN or FORCE_THTL) will be ignored.

Note: This register must not be used by systems with more than 1 logical processor, unless appropriate semaphoring software has been put in place to ensure that all threads/ processors are ready for the C2 state when the read to this register occurs.

Table 27-16. Offset 14h: LV2 - Level 2 Register

Description:					
View: PCI	**BAR:** PMBASE (IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 14h **Offset End:** 14h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7 :0	LV2Reg	See description above		00h	RO

27.3.3.7 Offset 28h: GPE0_STS - General Purpose Event 0 Status Register

Note: This register is symmetrical to the General Purpose Event 0 Enable Register. Unless indicated otherwise below, if the corresponding _EN bit is set, then when the STS bit is set, a Wake Event is generated. Once back in an S0 state (or if already in an S0 state when the event occurs), CMI will also generate an SCI if the SCIEN (PMBASE + 04h, bit 0) bit is set, or an SMI# if the SCIEN bit is not set. Bits 31:16 are reset by a CF9h write; bits 15:0 are not be reset by CF9 write. Bits 31:0 are reset by RSMRST#.

Table 27-17. Offset 28h: GPE0_STS - General Purpose Event 0 Status Register (Sheet 1 of 4)

Description:					
View: PCI	**BAR:** PMBASE (IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 28h **Offset End:** 28h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	GPIn_STS	0 = Software clears this bit by writing a 1 to it. 1 = These bits are set any time the corresponding GPIO is set up as an input and the corresponding GPIO signal is high (or low if the corresponding GP_INV bit is set). If the corresponding enable bit is set in the GPE0_EN register, then when the GPI[n]_STS bit is set: • If the system is in an S1, S3, S4 or S5 state, the event will also wake the system. • If the system is in an S0 state (or upon waking back to an S0 state), an SCI will be caused, depending on the GPI_ROUT bits for the corresponding GPI.		0h	RWC
15	Reserved	Reserved		0h	
14	Reserved	Reserved.		0h	

Table 27-17. Offset 28h: GPE0_STS - General Purpose Event 0 Status Register (Sheet 2 of 4)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 28h **Offset End:** 28h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13	PME_B0_STS	***Note:*** This bit will be set to 1 when any internal device with PCI Power Management capabilities on bus 0 asserts the equivalent of the PME# signal. Additionally, if the PME_B0_EN bit is set, and the system is in an S0 state, then the setting of the PME_B0_STS bit will generate an SCI (or SMI# if SCI_EN is not set). If the PME_B0_STS bit is set, and the system is in an S1/S3/S4 state (or S5 state due to SLP_TYP and SLP_EN), then the setting of the PME_B0_STS bit will generate a wake event, and an SCI (or SMI# if SCI_EN is not set) will be generated. If the system is in an S5 state due to power button override, then the PME_B0_STS bit will not cause a wake event or SCI. The default for this bit is 0. Writing a 1 to this bit position clears this bit		0h	RWC
12	Reserved	Reserved		0h	
11	PME_STS	This bit will be set to 1 by hardware when the PME# signal goes active. [Note CMI might be the cause of PME# going active in some cases]. Additionally, if the PME_EN bit is set, and system is in an S0 state, then the setting of the PME_STS bit will generate an SCI (or SMI# if SCI_EN is not set). If the PME_EN bit is set, and the system is in an S1/S3/S4 state (or S5 state due to SLP_TYP and SLP_EN), then the setting of the PME_STS bit will generate a wake event, and an SCI (or SMI# if SMI_EN is not set) will be generated. If the system is in an S5 state due to power button override, then PME_STS will not cause a wake event or SCI. This bit is cleared by writing a 1 to this bit position.		0h	RWC
10	Reserved	Reserved		0h	

Table 27-17. Offset 28h: GPE0_STS - General Purpose Event 0 Status Register (Sheet 3 of 4)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 28h **Offset End:** 28h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
09	PCI_EXP_STS	0 = Software clears this bit by writing a 1 to it. 1 = Set to 1 by hardware to indicate that: • The PME event message was received on one or more of the PCI Express* Ports • An Assert PMEGPE message received from the IMCH via NSI ***Notes:*** 1. Software attempts to clear this bit by writing a 1 to this bit position. If the PCI_EXP_STS bit went active due to an Assert PMEGPE message, then a Deassert PMEGPE message must be received prior to the software write in order for the bit to be cleared. 2. If the bit is not cleared and the corresponding PCI_EXP_EN bit is set, the level-triggered SCI will remain active. 3. A race condition exists where the PCI Express* device sends another PME message because the PCI Express* device was not serviced within the time when it must resend the message. This may result in a spurious interrupt, and this is comprehended and approved by the *PCI Express* Specification*. The window for this race condition is approximately 95-105 ms.		0h	RWC
08	RI_STS	0 = Software clears this bit by writing a 1 to it. 1 = Set by hardware when the RI# input signal goes active. The value of this bit is maintained through a G3 state.		0h	RWC
07	SMB_WAK_STS	0 = Wake event not caused by the SMBus logic. 1 = Set by hardware to indicate that the wake event was caused by the SMBus logic. ***Note:*** 1. If SMB_WAK_STS is set due to SMBus slave receiving a message, it will be cleared by internal logic when CPUTHRMTRIP event happens or by a Power Button Override event. However, CPUTHRMTRIP or Power Button override event will not clear SMB_WAK_STS if it was set due to SMBALERT# signal going active. 2. The SMBus controller will independently cause an SMI# so this bit does not need to do so (unlike the other bits in this register). 3. This bit is set by the SMBus slave command 01h (Wake/SMI#) even when the system is in the S0 state. Therefore, to avoid an instant wake on subsequent transitions to sleep states, software must clear this bit after each reception of the Wake/SMI# command or just prior to entering the sleep state. 4. The SMBALERT_STS bit (D31:F3:I/O Offset 00h:bit 5) must be cleared by software before clearing this bit.		0h	RWC

Table 27-17. Offset 28h: GPE0_STS - General Purpose Event 0 Status Register (Sheet 4 of 4)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 28h **Offset End:** 28h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	TCOSCI_STS	0 = TCO logic did Not cause SCI. 1 = Set by hardware when the TCO logic causes an SCI. This bit can be reset by writing a one to this bit position.		0h	RWC
05	Reserved	Reserved		0h	
04	Reserved	Reserved		0h	
03	USB1_STS	0 = USB UHCI controller 1 does Not need to cause a wake. 1 = Set by hardware when USB UHCI controller 1 needs to cause a wake. Wake event will be generated if the corresponding USB1_EN bit is set. This bit is only set by hardware and can be reset by writing a one to this bit position or a resume-well reset. Note: There is no support for wake from USB when in S3/S4/S5.		0h	RWC
02	Reserved	Reserved.		0h	
01	Reserved	Reserved		0h	
00	PROCHOT_STS	0 = PROCHOT# signal Not driven active as defined by the PROCHOT_POL bit 1 = Set by hardware anytime the PROCHOT# signal is driven active as defined by the PROCHOT_POL bit. Additionally, if the PROCHOT_EN bit is set, then the setting of the PROCHOT_STS bit will also generate a power management event (SCI or SMI#). This bit is cleared by S/W writing a one to this bit position or a resume-well reset.		0h	RWC

27.3.3.8 Offset 2Ch: PMBASE_GPE0_EN - General Purpose Event 0 Enables Register

Note: This register is symmetrical to the General Purpose Event 0 Status Register. All the bits in this register must be cleared to 0 based on a Power Button Override-CPU Thermal Trip event, or internal thermal sensor catastrophic condition. Unless otherwise noted, all bits are in the resume well. The Resume well bits are all cleared by RSMRST# and RTC well bits are cleared by RTEST#.

Table 27-18. Offset 2Ch: PMBASE_GPE0_EN - General Purpose Event 0 Enables Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 2Ch **Offset End:** 2Ch	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	GPIn_EN	These bits enable the corresponding GPI[n]_STS bits being set to cause an SCI and/or wake event. These bits are cleared by RSMRST#. ***Note:*** Mapping is as follows: bit 31 corresponds to GPI[15]... and bit 16 corresponds to GPI:[0].		0h	RW
15	Reserved	Reserved		0h	
14	Reserved	Reserved		0h	
13	PME_B0_EN	0 = Disable 1 = Enables the setting of the PME_B0_STS bit to generate a wake event and/or an SCI or SMI#. PME_B0_STS can be a wake event from the S1/S3/S4 state, or from S5 (if entered via SLP_TYP and SLP_EN) or power failure, but not Power Button Override. This bit defaults to 0. ***Note:*** It is only cleared by Software or RTEST#. It is not cleared by CF9h writes. This bit is in the RTC well.		0h	RW
12	Reserved	Reserved		0h	
11	PME_EN	0 = Disable. 1 = Enables the setting of the PME_STS to generate a wake event and/or an SCI. PME# can be a wake event from the S1/S3/S4 state or from S5 (if entered via SLP_EN, but not power button override). This bit is only cleared by software or RTEST#. It is not cleared by CF9h writes. This bit is in the RTC well.		0h	RW
10	Reserved	Reserved		0h	
09	PCI_EXP_EN	0 = Disable SCI generation upon PCI_EXP_STS bit being set. 1 = Enables an SCI when PCI_EXP_STS bit is set. This is used to allow the PCI Express ports, including the link to the IMCH, to cause an SCI due to wake/PME events.		0h	RW
08	RI_EN	When RI_EN and RI_STS are both set, a Wake event will occur. If RI_EN is not set, then when RI_STS is set, no Wake event will occur. 0 = Disable. 1 = Enables the setting of the RI_STS to generate a wake event. This bit is only cleared by software or RTEST#. This bit is in the RTC well.		0h	RW
07	Reserved	Reserved		0h	
06	TCOSCI_EN	When TCOSCI_EN and TCOSCI_STS are both set, an SCI will be generated. 0 = Disable. 1 = Enables the setting of the TCOSCI_STS to generate an SCI. This bit is in the resume well. This bit is only cleared by software or RSMRST#. It is not cleared by CF9h writes.		0h	RW
05	Reserved	Reserved		0h	

Table 27-18. Offset 2Ch: PMBASE_GPE0_EN - General Purpose Event 0 Enables Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 2Ch **Offset End:** 2Ch	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Resume[a]	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	Reserved	Reserved		0h	
03	USB1_EN	0 = Disable. 1 = Enables the setting of the USB1_STS to generate a wake event. The USB1_STS bit is set anytime USB controller 1 signals a wake event. Break events are handled via the USB interrupt. Note: There is no support for wake from USB when in S3/S4/S5.		0h	RW
02	PROCHOT_POL	This bit controls the polarity of the PROCHOT# pin needed to set the PROCHOT_STS bit. 0 = Low value on the PROCHOT# signal will set the PROCHOT_STS bit. 1 = HIGH value on the PROCHOT# signal will set the PROCHOT_STS bit.		0h	RW
01	Reserved	Reserved		0h	
00	PROCHOT_EN	0 = Disable. 1 = Active assertion of the PROCHOT# signal (as defined by the PROCHOT_POL bit) will set the PROCHOT_STS bit and generate a power management event (SCI or SMI).		0h	RW

a. Bits 0-7: Resume; Bits 8, 10-11, 13: RTC; Bits 9, 12, 16-31: Resume

27.3.3.9 Offset 30h: SMI_EN - SMI Control and Enable Register

Note: This register is symmetrical to the SMI Status Register.

Table 27-19. Offset 30h: SMI_EN - SMI Control and Enable Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 30h **Offset End:** 30h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 19	Reserved	Reserved		0h	
18	INTEL_USB2_EN	0 = Disables Intel-Specific USB 2.0 SMI logic. 1 = Enables Intel-Specific USB 2.0 SMI logic to cause SMI#.		0h	RW
17	LEGACY_USB2_EN	0 = Disable 1 = Enables legacy USB 2.0 logic to cause SMI#.		0h	RW
16 : 15	Reserved	Reserved		0h	

Table 27-19. Offset 30h: SMI_EN - SMI Control and Enable Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 30h **Offset End:** 30h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
14	PERIODIC_EN	0 = Disable 1 = Enables an SMI# to be generated when the PERIODIC_STS bit (PMBASE + 34h, bit 14) is set in the SMI_STS register (PMBASE + 34h).		0h	RW
13	TCO_EN	0 = Disables TCO logic generating an SMI#. 1 = Enables the TCO logic to generate SMI#. See Chapter 32.0, "High Precision Event Timers" for more details on TCO functions. If the NMI2SMI_EN bit is set, then SMIs that are caused by NMIs (i.e., rerouted) will not be gated by the TCO_EN bit. Even if the TCO_EN bit is 0, the NMIs will still be routed to cause the SMI#. ***Note:*** This bit can not be written once the TCO_LOCK bit (at offset 08h of TCO I/O Space) is set. This prevents unauthorized software from disabling the generation of TCO-based SMIs.		0h	RW
12	Reserved	Reserved		0h	
11	MCSMI_EN	0 = Disable. 1 = Enables IICH to trap accesses to the microcontroller range (62h or 66h) and generate an SMI#. Note that "trapped' cycles will be claimed by the IICH on PCI, but not forwarded to LPC.		0h	RW
10 : 08	Reserved	Reserved		0h	
07	BIOS_RLS	0 = This bit will always return 0 on reads. Writes of 0 to this bit have no effect. 1 = Enables the generation of an SCI interrupt for ACPI software when a one is written to this bit position by BIOS software. ***Note:*** GBL_STS being set will cause an SCI, even if the SCI_EN bit is not set. Software must take great care not to set the BIOS_RLS bit (which causes GBL_STS to be set) if the SCI handler is not in place.		0h	WO
06	SWSMI_TMR_EN	0 = Disable. Clearing the SWSMI_TMR_EN bit before the timer expires will reset the timer and the SMI# will not be generated. 1 = Starts Software SMI# Timer. When the SWSMI timer expires (the timeout period depends upon the SWSMI_RATE_SEL bit setting), SWSMI_TMR_STS is set and an SMI# is generated. SWSMI_TMR_EN stays set until cleared by software.		0h	RW
05	APMC_EN	0 = Writes to the APM_CNT register will not cause an SMI#. 1 = Enables writes to the APM_CNT register to cause an SMI#		0h	RW

Table 27-19. Offset 30h: SMI_EN - SMI Control and Enable Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 30h **Offset End:** 30h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	SMI_ON_SLP_EN	0 = Disables the generation of SMI# on SLP_EN. Note that this bit must be 0 before the software attempts to transition the system into a sleep state by writing a 1 to the SLP_EN bit. 1 = A write of 1 to the SLP_EN bit (bit 13 in PM1_CNT register) will generate an SMI#, and the system will not transition to the sleep state based on that write to the SLP_EN bit. This allows the SMI# handler work around chip-level bugs. It is expected that the SMI# handler will turn off the SMI_ON_SLP_EN bit before actually setting the SLP_EN bit.		0h	RW
03	LEGACY_USB_EN	0 = Disables legacy USB circuit 1 = Enables legacy USB circuit to cause SMI#.		0h	RW
02	BIOS_EN	0 = Disables the generation of SMI# when ACPI software writes a 1 to the GBL_RLS bit. 1 = Enables the generation of SMI# when ACPI software writes a 1 to the GBL_RLS bit.		0h	RW
01	EOS	End of SMI. This bit controls the arbitration of the SMI signal to the processor. This bit must be set in order to assert SMI# low to the processor after SMI# has been asserted previously. 0 = Once SMI# low is asserted, the EOS bit is automatically cleared. 1 = In the SMI handler, the processor must clear all pending SMIs (by servicing them and then clearing their respective status bits), set the EOS bit, and exit SMM. This will allow the SMI arbiter to reassert SMI upon detection of an SMI event and the setting of a SMI status bit. The SMI# signal will go inactive for 4 PCI clocks.		0h	RW
00	GBL_SMI_EN	0 = No SMI# will be generated. 1 = Enables the generation of SMIs in the system upon any enabled SMI event. This bit is reset by a PCI reset event. ***Note:*** When the SMI_LOCK bit is set, this bit cannot be changed.		0h	RW

27.3.3.10 Offset 34h: SMI_STS - SMI Status Register

Note: If the corresponding _EN bit is set when the _STS bit is set, CMI will cause an SMI# (except bits 8-10, which do not cause SMI#.)

Table 27-20. Offset 34h: SMI_STS - SMI Status Register (Sheet 1 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 27	Reserved	Reserved		0h	
26	SPI_SMI_STS	This bit will be set when the SPI logic is requesting an SMI#.		0	RO
25 : 21	Reserved	Reserved		0h	
20	Reserved	Reserved.		0h	
19	Reserved	Reserved		0h	
18	INTEL_ USB2_STS	This non-sticky read-only bit is a logical OR of each of the SMI status bits in the Intel-Specific USB 2.0 SMI Status Register ANDed with the corresponding enable bits. This bit will not be active if the enable bits are not set. Writes to this bit will have no effect.		0h	RO
17	LEGACY_ USB2_STS	This non-sticky read-only bit is a logical OR of each of the SMI status bits in the USB 2.0 Legacy Support Register ANDed with the corresponding enable bits. This bit will not be active if the enable bits are not set. Writes to this bit will have no effect.		0h	RO
16	SMBUS_ SMI_STS	This bit is set to 1 to indicate that the SMI# was caused by: A. The SMBus Slave receiving a message that an SMI# must be caused. B. The SMBALERT# signal goes active and the SMB_SMI_EN bit is set and the SMBALERT_DIS bit is cleared. C. The SMBus Slave receiving a HOST_NOTIFY message and the HOST_NOTIFY_INTREN and the SMB_SMI_EN bits are set. D. The SMBus Slave receiving a “SMI in S0” message. This bit is sticky. It is cleared by writing a 1 to this bit position. ***Note:*** This bit is set from the 64 KHz clock domain used by the SMBus. Software must wait at least 15.63 µs (= 1/64 kHz) after the initial assertion of this bit before clearing it.		0h	RWC
15	SERIRQ_ SMI_STS	0 = SMI# not caused by SERIRQ decoder. 1 = Indicates the SMI# was caused by the SERIRQ decoder. ***Note:*** This bit is **not** sticky. Writes to this bit will have no effect.		0h	RO
14	PERIODIC_STS	This bit will be set at the rate determined by the PER_SMI_SEL bits. If the PERIODIC_EN bit is also set, an SMI# is generated. This bit is cleared by writing a 1 to this bit position.		0h	RWC
13	TCO_STS	0 = SMI not caused by TCO logic. 1 = Indicates SMI was caused by the TCO logic. The reset value of this bit may be overwritten soon after reset by the TCO counter. Hence, it is possible that a read to this register after reset will yield a 1 in this field. ***Note:*** Will not cause wake event. This bit is cleared by writing a 1 to this bit position.		0h	RWC

Table 27-20. Offset 34h: SMI_STS - SMI Status Register (Sheet 2 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
12	DEVMON_STS	This read-only bit is set when bit 0 in the DEVTRAP_STS register is set. It is not sticky, so writes to this bit will have no effect. See Section 27.3.1.3.]		0h	RO
11	MCSMI_STS	0 = Indicates that there has been no access to the power management microcontroller range (62h or 66h). 1 = Set if there has been an access to the power management microcontroller range (62h or 66h) and the Microcontroller Decode Enable #1 bit in the LPC Bridge I/O Enables configuration register is 1. Note that this implementation assumes that the Microcontroller is on LPC, if this changes in the future (i.e. PCI e-based SIO), then the implementation will need to remove the LPC Decode Enable dependency. If this bit is set, and the MCSMI_EN bit is also set, CMI will generate an SMI#. This bit is set by hardware and cleared by software writing a 1 to its bit position.		0h	RWC
10	GPE1_STS	This bit is a logical OR of the bits in the ALT_GPI_SMI_STS register that are also set up to cause an SMI# (as indicated by the GPI_ROUT registers) and have the corresponding bit set in the ALT_GPI_SMI_EN register. 0 = SMI# was not generated by a GPI assertion. 1 = SMI# was generated by a GPI assertion. Bits that are not routed to cause an SMI# will have no effect on this bit. This bit is NOT sticky. Writes to this bit will have no effect.		0h	RO
09	GPE0_STS	This bit is a logical OR of bits 13, 11, 8:6, 4:3 and 0 in the GPE0_STS register (PMBASE + 28h) that also have the corresponding bit set in the GPE0_EN register (PMBASE + 2Ch). This bit is NOT sticky. 0 = SMI# was not generated by a GPE0 event. 1 = SMI# was generated by a GPE0 event. ***Note:*** Writes to this bit will have no effect. The setting of this bit does not cause the SMI# ***Note:*** Bits 31:16 of the GPE0_STS register are not capable of generating SMIs; therefore, they do not set this SMI status bit.		0h	RO
08	PM1_STS_REG	This is an OR of the bits (except for bits 5 and 4) in the ACPI PM1 Status Reg. (offset PMBASE+00h). Not sticky. Writes to this bit have no effect. 0 = SMI# was not generated by a PM1_STS event. 1 = SMI# was generated by a PM1_STS event. This bit gets set when PM1_STS.TMROF_STS gets set. Hence, it is highly likely that a read to this register after reset will yield a 1 in this field. ***Note:*** The setting of this bit does not cause the SMI#.		0h	RO
07	Reserved	Reserved.		0h	
06	SWSMI_ TMR_STS	0 = Software SMI# Timer has Not expired. 1 = Set by the hardware when the Software SMI# Timer expires. This bit will remain 1 until the software writes a 1 to this bit.		0h	RWC

Table 27-20. Offset 34h: SMI_STS - SMI Status Register (Sheet 3 of 3)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	APM_STS	SMI# was generated by a write access to the APM control register and if the APMC_EN bit is set. 0 = No SMI# generated by write access to APM Control register with APMCH_EN bit set. 1 = SMI# was generated by a write access to the APM Control register with the APMC_EN bit set. This bit is cleared by writing a one to its bit position.		0h	RWC
04	SMI_ON_SLP_EN_STS	This bit will be set when a write access attempts to set the SLP_EN bit. 0 = No SMI# caused by write of 1 to SLP_EN bit when SLP_SMI_EN bit is also set. 1 = Indicates an SMI# was caused by a write of 1 to SLP_EN bit when SLP_SMI_EN bit is also set. This bit is cleared by writing a 1 to this bit position.		0h	RWC
03	LEGACY_USB_STS	This non-sticky read-only bit is a logical OR of each of the SMI status bits in the USB Legacy Keybd Register ANDed with the corresponding enable bits. This bit will not be active if the enable bits are not set. 0 = SMI# was not generated by USB Legacy event. 1 = SMI# was generated by USB Legacy event.		0h	RO
02	BIOS_STS	0 = No SMI# generated due to ACPI software requesting attention. 1 = SMI# was generated due to ACPI software requesting attention (writing a 1 to the GBL_RLS bit with the BIOS_EN bit set).		0h	RWC
01 : 00	Reserved	Reserved		0h	

27.3.3.11 Offset 38h: ALT_GPI_SMI_EN - Alternate GPI SMI Enable Register

Table 27-21. Offset 38h: ALT_GPI_SMI_EN - Alternate GPI SMI Enable Register

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 38h **Offset End:** 38h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	ALT_GPI_SMI_EN	These bits are used to enable the corresponding GPIO to cause an SMI#. In order for these bits to have any effect, the following must be true. 1. The corresponding bit in the ALT_GPI_SMI_EN register is set. 2. The corresponding GPI must be routed in the GPI_ROUT register to cause an SMI. 3. The corresponding GPIO must be implemented. All bits are in the resume well.		0000h	RW

27.3.3.12 Offset 3Ah: ALT_GPI_SMI_STS - Alternate GPI SMI Status Register

Table 27-22. Offset 3Ah: ALT_GPI_SMI_STS - Alternate GPI SMI Status Register

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 3Ah **Offset End:** 3Ah	
Size: 16 bit	**Default:** 0000h			**Power Well:** Resume	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	ALT_GPI_ SMI_STS	These bits report the status of the corresponding GPIs. 1 = active, -0 = inactive. These bits are sticky. If the following conditions are true, then an SMI# will be generated and the ALT_GPI_SMI_STS bit set: 1. The corresponding bit in the ALT_GPI_SMI_EN register is set 2. The corresponding GPI must be routed in the GPI_ROUT register to cause an SMI. 3. The corresponding GPIO must be implemented. All bits are in the resume well. Default for these bits are dependent on the state of the GPI pins.		0000h	RWC

27.3.3.13 Offset 44h: DEVTRAP_STS - DEVTRAP_STS Register

Each bit indicates if an access has occurred to the corresponding devices trap range, or for bits 6:9 if the corresponding PCI interrupt is active. Write 1 to the same bit position to clear it. This register is used by APM power management software to see if there has been system activity. The periodic SMI# timer indicates if it is the right time to read the DEVTRAP_STS register (PMBASE + 44h).

Table 27-23. Offset 44h: DEVTRAP_STS - DEVTRAP_STS Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)	**Bus:Device:Function:** 0:31:0		**Offset Start:** 44h **Offset End:** 44h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 13	Reserved	Reserved		0h	
12	D12_TRP_STS	**KBC (60/64h):** 0 = Indicates that there has been no access to this device's I/O range. 1 = This device's I/O range has been accessed. Clear this bit by writing a 1 to the bit location.		0h	RWC
11 : 10	Reserved	Reserved		0h	
09	D9_TRP_STS	**PIRQ[D or H]:** 0 = The corresponding PCI interrupts have not been active. 1 = At least one of the corresponding PCI interrupts has been active. Clear this bit by writing a 1 to the bit location.		0h	RWC

Table 27-23. Offset 44h: DEVTRAP_STS - DEVTRAP_STS Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** PMBASE (IO)		**Bus:Device:Function:** 0:31:0	**Offset Start:** 44h **Offset End:** 44h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	D8_TRP_STS	**PIRQ[C or G]:** 0 = The corresponding PCI interrupts have not been active. 1 = At least one of the corresponding PCI interrupts has been active. Clear this bit by writing a 1 to the bit location.		0h	RWC
07	D7_TRP_STS	**PIRQ[B or F]:** 0 = The corresponding PCI interrupts have not been active. 1 = At least one of the corresponding PCI interrupts has been active. Clear this bit by writing a 1 to the bit location.		0h	RWC
06	D6_TRP_STS	**PIRQ[A or E]:** This bit will be set if PCI IRQ A or PCI IRQ E goes active (by the pin or internal signal). 0 = The corresponding PCI interrupts have not been active. 1 = At least one of the corresponding PCI interrupts has been active. Clear this bit by writing a 1 to the bit location.		0h	RWC
05 : 00	Reserved	Reserved		0h	

27.4 SMI#/SCI Generation

Upon any SMI# event taking place, IICH will assert SMI# to the processor, which will cause it to enter SMM space. SMI# remains active until the EOS bit is set. When the EOS bit is set, SMI# will go inactive for a minimum of 4 PCICLK. If another SMI event occurs, SMI# will be driven active again.

The SCI is a level-mode interrupt that is typically handled by an ACPI-aware operating system. In non-APIC systems (which is the default), the SCI IRQ is routed to one of the 8259 interrupts (IRQ9, 10, or 11). The 8259 interrupt controller must be programmed to level mode for that interrupt.

In systems using the APIC, the SCI can be routed to interrupts 9, 10, 11, 20, 21, 22, or 23. The interrupt polarity changes depending on whether it is on an interrupt shareable with a PIRQ or not; see Section 27.3.1 for details. The interrupt will remain asserted until all SCI sources are removed.

Table 27-24 shows which events can cause an SCI and Table 27-26 shows the causes of an SMI#.

Note: Some events can be programmed to cause either an SMI# or SCI. The usage of the event for SCI (instead of SMI#) is typically associated with an ACPI-based system.

Table 27-24. Causes of SCI

Cause	Additional Enables (See Note 1)	Where Reported
PME#	PME_EN = 1	PME_STS
Internal EHCI wake (PME_B0)	PME_B0_EN = 1	PME_B0_STS
Power Button Press	PWRBTN_EN = 1	PWRBTN_STS
RTC Alarm	RTC_EN = 1	RTC_STS
Ring Indicate	RI_EN = 1	RI_STS
USB #1 wakes	USB1_EN = 1	USB1_STS
PROCHOT# pin active		
ACPI Timer overflow (2.34 seconds)	TMROF_EN = 1	TMROF_STS
Any GPI	GPI[x]_Route = 10, GPE0[x]_EN = 1	GPE0[x]_STS
TCO SCI Logic (see Table 27-25)	TCOSCI_EN = 1	TCOSCI_STS
BIOS_RLS written to 1	GBL_EN = 1	GBL_STS

Notes: Causes of SCI:
1. SCI_EN must be 1 to enable SCI
2. SCI can be routed to cause Interrupt 9:11 or 20:23 (20:23 only available in APIC mode)

Note: There are various sources that cause the TCO SCI, shown in Table 27-25.

Table 27-25. Causes of TCO SCI

Cause	Additional Enables	Where Reported
Message from IMCH	None	IMCHSCI_STS

Table 27-26. Causes of SMI# (Sheet 1 of 2)

Cause	Additional Enables	Where Reported	Synch
PME#	SCI_EN = 0, PME_EN = 1	PME_STS	
Internal EHCI wake (PME_B0)	SCI_EN = 0, PME_B0_EN = 1	PME_B0_STS	
Power Button Press	SCI_EN = 0, PWRBTN_EN = 1	PWRBTN_STS	
RTC Alarm	SCI_EN = 0, RTC_EN = 1	RTC_STS	
Ring Indicate	SCI_EN = 0, RI_EN = 1	RI_STS	
USB #1 wakes	SCI_EN = 0, USB1_EN = 1	USB1_STS	
PROCHOT# pin active	SCI_EN = 0		
ACPI Timer overflow (2.34 seconds)	SCI_EN = 0, TMROF_EN = 1	TMROF_STS	
Any GPI	GPI[x]_Route = 01, ALT_GPI_SMI[x]_EN = 1	GPE1_STS, ALT_GPI_SMI[x]_STS	
TCO SMI Logic	TCO_EN = 1	TCO_STS	
NMI I (and NMIs mapped to SMI) See NMI section for causes.	NMI2SMI_EN = 1	TCO_STS, NMI2SMI_STS	
GBL_RLS written to 1	BIOS_EN = 1	BIOS_STS	X
Write to B2h register	APMC_EN	APM_STS	X
Periodic timer expires	PERIODIC_EN = 1	PERIODIC_STS	
64 ms timer expires	SWSMI_TMR_EN = 1	SWSMI_TMR_STS	
Internal Thermal Throttle		INTERNAL_TT_STS	
Monitor Status		MONITOR_STS	X
Enhanced USB Legacy Support Event	LEGACY_USB2_EN = 1	LEGACY_USB2_STS	
Enhanced USB Intel-Specific Event	INTEL_USB2_EN = 1	INTEL_USB2_STS	
Classic USB Legacy logic (Port 64/60 rd/wr, End of pass-through)	LEGACY_USB_EN = 1	LEGACY_USB_STS	X
Classic USB Legacy logic (IRQ)	LEGACY_USB_EN = 1	LEGACY_USB_STS	
Serial IRQ SMI Reported	None	SERIRQ_SMI_STS	
Device Monitors (D15:D0)matches an address in its range	See Trap Section 18.2.2.1.	DEVMON_STS,	X
SMBus Host Controller	SMB_SMI_EN, Host Controller enabled	Various bits in the SMBus Host Status Register	
SMBus Slave SMI message	None	SMBUS_SMI_STS	
SMBus SMBALERT# signal active	None	SMBUS_SMI_STS	
SMBus Host Notify message received	HOST_NOTIFY_INTREN	SMBUS_SMI_STS, HOST_NOTIFY_STS	

Table 27-26. Causes of SMI# (Sheet 2 of 2)

Cause	Additional Enables	Where Reported	Synch
Access to Microcontroller Range (62h/66h)	MCSMI_EN	MCSMI_STS	X
SLP_EN bit written to 1	SMI_ON_SLP_EN = 1	SMI_ON_SLP_EN_STS	X

Notes: Causes of SMI#:
1. GBL_SMI_EN must be 1 to enable SMI
2. EOS must be written to 1 to reenable SMI for the next one
3. Some SMI#s are considered "synchronous", in that the processor must recognize the SMI# prior to completing the instruction (I/O read, I/O write, Memory read, or Memory write) that must cause the SMI#. This is accomplished by having the SMI# signal go active to the processor prior to the processor observing the RDY# signal that terminates the cycle. SMI#s marked with X in the Synch column are treated as Synchronous.
4. NMI2SMI_STS isn't gated by TCO_EN.
5. An SMI# must be fully enabled when CMI is also enabled to trap cycles. If SMI# is not enabled in conjunction with the trap enabling, then hardware behavior is undefined. [Note: added as per SUN# 47], part 3.

Table 27-27. Causes of TCO SMI#

Cause	Additional Enables	Where Reported
Year 2000 Rollover	None	NEWCENTURY_STS
TCO TIMEROUT	None	TIMEOUT
OS writes to TCO_DAT_IN register	None	OS_TCO_SMI
NMI occurred (and NMIs mapped to SMI)	NMI2SMI_EN = 1	NMI2SMI_STS
Note: NMI2SMI_STS isn't gated by TCO_EN.		
INTRUDER# signal goes active	INTRD_SEL = 10	INTRD_DET
Changes of the BIOSWP bit from 0 to 1	BLD = 1	BIOSWR_STS
Message from IMCH		IMCHSMI_STS
Write attempted to BIOS	BIOSWP = 1	BIOSWR_STS

See Chapter 32.0, "High Precision Event Timers" for details on the TCO SMI# causes.

27.4.0.1 PCI Express* SCI

PCI Express* ports and the IMCH (via NSI) have the ability to cause PME using messages. When a PME message is received, the PCI_EXP_STS bit is set. If the PCI_EXP_EN bit is also set, CMI can cause an SCI via the GPE1_STS register.

27.5 Dynamic Processor Clock Control

27.5.1 Overview

CMI has primary control for dynamically starting and stopping system clocks. The clock control is used for the transitions among the various S0/Cx states (i.e., IA 32 core throttling). Each dynamic clock control method is described in this section. The various Sleep states may also perform types of non-dynamic clock control, and are described in Section 27.4.

CMI supports the ACPI C0, C1 and C2 states

The Dynamic Clock control is handled using the following signals:

- STPCLK# - Used to halt the IA 32 core instruction stream

The C1 state is entered based on the IA 32 core performing an autohalt instruction.

The C2 state is entered based on the IA 32 core reading the Level 2 register.

The C1, and C2 states end due to a Break event. Based on the break event, CMI returns the system to a C0 state. Table 27-28 lists the possible break events from the C2, states.

Table 27-28. Break Events

Event	Breaks From	Comment
Any unmasked interrupt goes active	C2	IRQ[0:15] when using the 8259s, IRQ[0:23] for I/O (x) APIC. Since SCI is an interrupt, any SCI will also be a break event.
Any internal event that will cause an NMI or SMI#	C2	Many possible sources
Any internal event that will cause INIT# to go active	C2	Could be indicated by the keyboard controller via the RCIN input signal.
RTC Interrupt Pending	C2	Only available if the RTC Interrupt (IRQ8) is enabled as a break event (See RTC Interrupt Break Enable bit in Section 29.3.1.1).
CPU Pending Break Event Indication	C2	Only available if FERR# is enabled for break event indication (See FERR# MUX Enable bit in Section 29.3.1.1)
REQ-C0 message from IMCH	C2	Can be sent at any time after the Ack-C2 message and before the Ack-C0 message (i.e., any time not in C0 state).

The Pending Break Event (PBE) indication from the IA 32 core is supported using the FERR# signal. The following rules apply:

1. When STPCLK# is detected active by the IA 32 core, the FERR# signal from the IA 32 core will be redefined to indicate whether an interrupt is pending. The signal is active low (i.e., FERR# will be low to indicate a pending interrupt).
2. When the STPCLK# asserts, it will latch the current state of the FERR# signal and continue to present this state to the FERR# state machine (independent of what the FERR# pin does after the latching).
3. When the Stop-Grant cycle is detected, it will start looking at the FERR# signal as a break event indication. If FERR# is sampled low, a break event is indicated. This will force a transition to the C0 state.
4. When the IA 32 core detects the deassertion of STPCLK#, the IA 32 core will start driving the FERR# signal with the natural value (i.e., the value it would do if the pin

was not muxed). The time from STPCLK# inactive to the FERR# signal transitioning back to the native function must be less than 120 ns.

5. At least 180 ns passes after deasserting STPCLK# and then starts using the FERR# signal for an indication of a floating point error. The maximum time that may pass is bounded such that it must have a chance to look at the FERR# signal before reasserting STPCLK#. Based on current implementation, that maximum time would be 240 ns (8 PCI clocks). Since the IA 32 core has 120 ns to revert to the proper FERR# function, there are 60 ns of margin inherent in the timings.

The break event associated with this mechanism does not need to set any particular status bit, since the pending interrupt will be serviced by the processor after returning to the C0 state.

27.5.2 Transition Rules Among S0/Cx and Sx States

The following priority rules and assumptions apply among the various S0/Cx and throttling states:

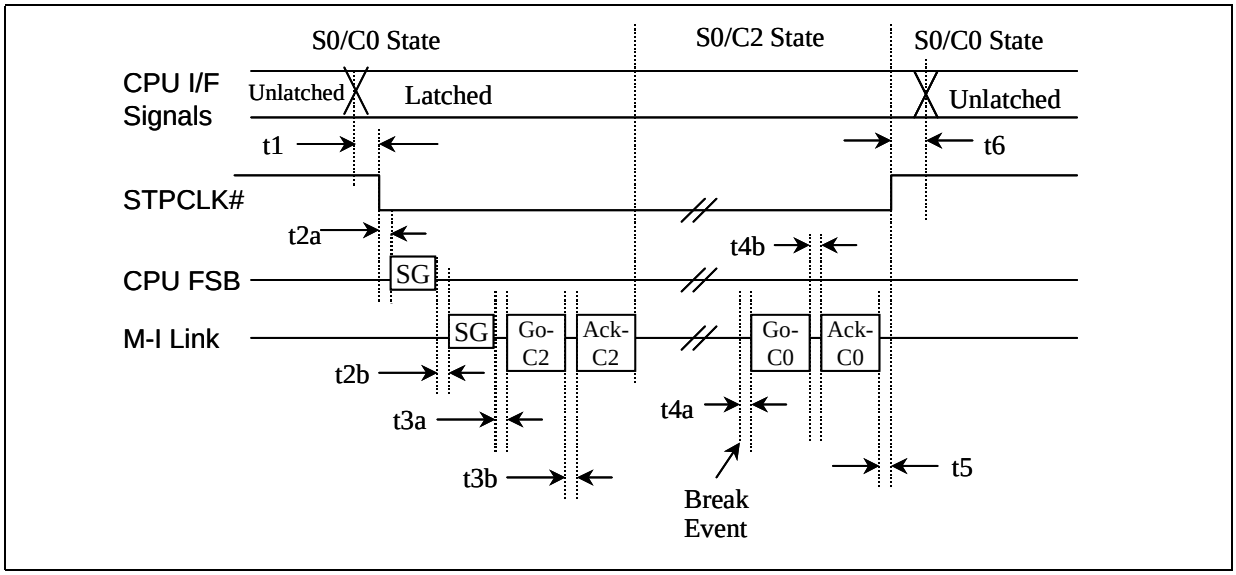
- Entry to any S0/Cx state is mutually exclusive with entry to S1, S3, S4 or S5 state. This is because the processor can only perform one register access at a time and Sleep states have higher priority than thermal throttling.
- ..When the SLP_EN bit is set (system going to a S1, S3, S4 or S5 sleep state), the THTL_EN and FORCE_THTL bits can be internally treated as being disabled (no throttling while going to sleep state).
- If the THTL_EN or FORCE_THTL bits are set, and a Level 2 read then occurs, the system must immediately go and stay in a C2 state until a break event occurs. A Level 2 read has higher priority than the software initiated throttling.
- After an exit from a C2 state (due to a Break event), and if the THTL_EN or FORCE_THTL bits are still set, the system will continue to throttle STPCLK#. The first transition on STPCLK# active can be delayed by up to one PROCHOT period (1024 PCI clocks = 30.72 µs), depending on the time of the break event.
- The IMCH (or equivalent) must post Stop-Grant cycles in such a way that the processor gets an indication of the end of the special cycle prior to CMI observing the Stop-Grant cycle. This ensures that the STPCLK# signal stays active for a sufficient period after the processor observes the response phase.
- If in the C1 state and the STPCLK# signal goes active, the processor will generate a Stop-Grant cycle, and the system must go to the C2-like state. When STPCLK# goes inactive, it must return to the C1 state.

27.5.3 S0/C0, S0/C2, Entry/Exit Timings and Sequences

The timings associated with the C0-C2-C0, sequences are shown in the following figures and tables.

27.5.3.1 C0→C2→C0 Timings and Diagram

Figure 27-1. C0→C2→C0 Entry/Exit Timings



Note: In Figure 27-1, the M-I Link must be labeled as “NSI” and the “SG” message on NSI must be “Req-C2”

27.5.3.2 C0→C2 Entry Sequence

Table 27-29. C0→C2→C0 Timings

Sym	Min	Max	Units	Description
T1	0	Note 1		CPU Interface Signals Latched prior to STPCLK# active. Note that this does not apply for synchronous SMI's. Changed as per DCN #014, part 1.
T2a	0	Note 2		STPCLK# active to Stop-Grant cycle on processor front-side bus (can wait forever)
T2b	0	Note 3		Stop-Grant on FSB to Stop-Grant on NSI. ***Note:*** This is according to an IMCH specification.
T3a	0	Note 1		Stop-Grant on NSI to Go-C2 message. This must be as short as feasible.
T3b	128	Note 3	BCLK	Go C-2 message to Ack-C2 message. Note that this is according to an IMCH specification and is only required if the IMCH has the CPULSLP# signal. It is needed to enforce the Stop-Grant to CPUSLP# timing requirements. If the IMCH does not have the CPUSLP# signal, then this can be 0.
T4a	0	8	PCI CLK	Break Event to when GO_C0 message is ready to be sent. The actual message may be delayed if NSI is busy with other traffic.
T4b	0	Note 3		Go-C0 message to Ack-C0 message. Note that this is an IMCH specification. This must be as short as feasible.
T5	0	Note 1		End of Ack_C0 message to STPCLK# high
T6	8	9	PCI CLK	STPCLK# high to processor interface signals unlatched.

1. This value must be small (a few PCI clocks). For messages, it may be difficult to determine the maximum time, since power management messages may have to wait for other traffic on NSI.
2. This is a processor specification that is unbounded.
3. This is an IMCH specification. The maximum is presently not specified. CMI should not be dependent on this specific value.

The processor goes from a C0 to a C2 state because all of the threads in the processor are idle and have nothing to do. The decision to go to the C2 state is made by software.

The following timings are shown in Figure 27-1.

1. The processor reads the Level 2 register.
2. t1 prior to asserting STPCLK#, CMI will latch the processor interface signals, except SMI# activation due to a synchronous SMI event. Changed as per DCN #014, part 3.
3. t2a later, in response to observing STPCLK# active, the processor(s) performs one or more Stop-Grant cycles on the front-side bus.
4. t2b later, IMCH forwards the last Stop-Grant cycle to CMI via NSI. This will be called “REQ-C2.”
5. The processor is now in C2 state. There are some additional steps below that must complete between the IICH and IMCH to keep them in synchronization.
6. t3a after receiving the Stop-Grant from the IMCH, the IICH sends a Go-C2 message to the IMCH
7. t3b after receiving the Go-C2 message, the IMCH sends an Ack-C2 message. At this point, the IMCH is permitted to send the REQ-C0 (Break-Ind) message.

27.5.3.3 C2→C0 Break Sequence

CMI returns the processor to a C0 state in order to execute code. This is due to a break event. See Table 27-28 for the various break event causes.

The following timings are shown in Figure 27-1.

1. A Break event is detected
2. t4a later, CMI sends GO_C0 message to the IMCH.
3. t4b after receiving the Go-C0 message, then IMCH sends an Ack-C0 to the IICH. At this point, the IMCH is not permitted to send the REQ-C0 (Break Ind) Message.
4. t5 after receiving the Ack-C0 message, CMI deasserts STPCLK# to the processor (this enables processor instruction stream)
5. The processor is now back in a C0 state
6. t6 after deasserting STPCLK#, CMI unlatches the processor interface signals, except SMI#, which was not latched for synchronous SMI events. Changed as per DCN #014, part 4.

27.6 Sleep States

27.6.1 Sleep State Overview

CMI directly supports different sleep states (S1, S3, S4 or S5), which are entered by setting the SLP_EN bit, or due to a Power Button press. The entry to the Sleep states are based on several assumptions:

- Entry to a Cx state is mutually exclusive with entry to a Sleep state. This is because the processor can only perform one register access at a time. A request to Sleep always has higher priority than throttling.
- Prior to setting the SLP_EN bit, the software turns off processor-controlled throttling. Thermal throttling cannot be disabled, but setting the SLP_EN bit disables thermal throttling (since S1, S3, S4 or S5 sleep states have higher priority).
- The G3 state cannot be entered via any software mechanism. The G3 state indicates a complete loss of power.

Table 27-30 shows the differences in the sleeping states with regard to the listed output signals:

27.6.2 Initiating Sleep States

Table 27-30. Sleep State Output Conditions

State	STPCLK#	CPUSLP#	SLP_S3#	SLP_S4#	SLP_S5#
S1	Active	Optionally Active	Inactive	Inactive	Inactive
S3	Active	Plane off	Active	Inactive	Inactive
S4	Active	Plane off	Active	Active	Inactive
S5	Active	Plane off	Active	Active	Active

Entry to Sleep states (S1, S3, S4 or S5) are initiated by any of the following methods:

1. Masking interrupts, turning off all bus master enable bits, setting the desired type in the SLP_TYP field and setting the SLP_EN bit. The hardware will then put the system into the corresponding Sleep state.
2. Pressing the PWRBTN# signal for more than four seconds to cause a Power Button Override event. In this case the transition to the S5 state will be less graceful, since there will be no dependencies on observing Stop-Grant cycles from the processor or on clocks other than the RTC clock.

Other Assumptions:

- Entry to a Cx state is mutually exclusive with entry to a Sleep state. This is because the processor can only perform one register access at a time. A request to Sleep always has higher priority than throttling.
- Setting the SLP_EN bit will disable all throttling (since S1, S3, S4 or S5 sleep states have higher priority).
- The G3 state cannot be entered via any software mechanism. The G3 state indicates a complete loss of power.
- Before entering sleep state, an ACPI OS will mask all interrupts and will turn off all bus master enable bits. For non-ACPI systems, the BIOS will mask interrupts and turn off all bus master enable bits. Interrupts might not be masked at the I/O subsystem. Some Operating Systems have been observed to only mask interrupts inside the processor.

Table 27-31. Sleep Types

Sleep Type	Comment
S1	CMI asserts the STPCLK# signal. It also has the option to assert CPUSLP# signal. This lowers the processor's power consumption. No snooping is possible in this state.
S3	CMI asserts SLP_S3#. The SLP_S3# signal controls the power to non-critical circuits. Power is only retained to devices needed to wake from this sleeping state, as well as to the memory.
S4	CMI asserts SLP_S3# and SLP_S4#. The SLP_S4# signal shuts off the power to the memory subsystem. Only devices needed to wake from this state should be powered.
S5	The SLP_S5# signal shuts off the power to the memory subsystem. Only devices needed to wake from this state must be powered. CMI asserts SLP_S3#, SLP_S4# and SLP_S5#.

27.6.3 Exiting Sleep States

Sleep states (S1, S3, S4 or S5) are exited based on Wake events. The Wake events will force the system to a full on state (S0), although some non-critical subsystems might still be shut and have to be brought back manually. For example, the hard disk may be shut during a sleep state, and have to be enabled via an I/O pin before it can be used.

Note: Upon exit from CMI-controlled Sleep states, the WAK_STS bit will be set. To enable Wake Events, the possible causes of wake events (and their restrictions) are shown in Table 27-32.

There is no support for wake from USB when in S3/S4/S5.

Table 27-32. Causes of Wake Events

Cause	States can wake from:	How Enabled
RTC Alarm	S1,S3,S4,S5	Set RTC_EN bit in PM1_EN Register
Power Button	S1,S3,S4,S5	Always enabled as Wake event
GPI[0:15]	S1,S3,S4,S5	GPE0_EN register (after having gone to S5 via SLP_EN, but not after a power failure.) GPIs that are in the core well are not capable of waking the system from sleep states where the core well is not powered.
Classic USB	S1	Set USB1_EN bits in GPE0_EN Register
RI	S1,S3,S4,S5	Set RI_EN bit in GPE0_EN Register
Primary PME#	S1,S3,S4,S5 (Note 2)	PME_B0_EN bit in GPE0_EN register
Secondary PME# (pin)	S1,S3,S4,S5 (Note 2)	PME_EN bit in GPE0_EN register.
SMBus ALERT# Signal	S1,S3,S4,S5	Always Enabled as a Wake Event
SMBus Slave Message	S1,S3,S4,S5, including S5-Power Button Override	Three SMBus commands always enabled as Wake events. These commands (see Note 1. below) can wake from S5 due to Power Button.
SMBus Host Notify message received	S1,S3,S4,S5	HOST_NOTIFY_WKEN bit SMBus Slave Command register. Reported in the SMB_WAK_STS bit in GPE0_STS register.

Notes:

1. If in the S5 state due to a powerbutton override or THRMTRIP#, the only wake events are Power Button, Wake SMBus Slave Message (01h), and Hard Reset SMBus Slave Messages (03h, 04h).
2. PME#, RTC, GPI[0:n], and RI# will be wake events from S5 only if it was entered via software setting the SLP_EN and SLP_TYP bits, or if there is a power failure.
3. GbE wake-up capability (Wake on LAN) is described in Section 37.5.10, “Wake on LAN” on page 1402
4. There is no support for wake from USB when in S3/S4/S5.

Table 27-33. GPI Wake Events

GPI	Power Well	Wake From	Notes
GPI[12, 11, 7:0]	Core	S1	ACPI Compliant
GPI[8]	Resume	S1,S3,S4,S5	ACPI Compliant

The latency to exit the various sleep states varies greatly and is heavily dependent on power supply design, so much so that the exit latencies due to CMI are insignificant.

27.6.4 Sx-G3-Sx, Handling Power Failures

In systems, power failures can occur if the AC power is cut (a real power failure) or if the system is unplugged. In either case, PWROK and RSMRST# are assumed to go low.

Depending on when the power failure occurs and how the system is designed, different transitions could occur due to a power failure.The AFTER_G3 bit provides the ability to program whether or not the system should boot once power returns after a power loss event. If the policy is to not boot, the system remains in an S5 state. There are only three possible events that will wake the system after a power failure.

1. **PWRBTN#:** PWRBTN# is always enabled as a wake event. When RSMRST# is low (G3 state), the PWRBTN_STS bit is reset. When CMI exits G3 after power returns

(RSMRST# goes high), the PWRBTN# signal is already high (because V_{CC}-standby goes high before RSMRST# goes high) and the PWRBTN_STS bit is 0.

2. **RI#:** RI# does not have an internal pull-up. Therefore, if this signal is enabled as a wake event, it is important to keep this signal powered during the power loss event. If this signal goes low (active), when power returns the RI_STS bit is set and the system interprets that as a wake event.
3. **RTC Alarm:** The RTC_EN bit is in the RTC well and is preserved after a power loss. Like PWRBTN_STS the RTC_STS bit is cleared when RSMRST# goes low.

CMI monitors both PWROK and RSMRST# to detect for power failures. If PWROK goes low, the PWROK_FLR bit is set. If RSMRST# goes low, PWR_FLR is set. Software can clear PWR_FLR by writing a 1 to that bit

Although PME_EN is in the RTC well, this signal cannot wake the system after a power loss. PME_EN is cleared by RTEST#, and PME_STS is cleared by RSMRST#.

Table 27-34. Transitions Due To Power Failure

State at Power Failure	AFTERG3_EN bit	Transition When Power Returns
S0, S1, S3	1 0	S5 S0
S4	1 0	S4 S0
S5	1 0	S5 S0

The power failure bit (PWR_FLR) is set after any power failure. Software can clear it by writing a 1 to that bit.

The CPUPWR_FLR bit separately reports power failures that result in VRMPWRGD going inactive.

27.7 Processor Thermal Management

CMI has several mechanisms to assist with managing thermal problems in the system.

27.7.1 PROCHOT# Signal for SMI# or SCI

The PROCHOT# signal is used as a status input from a thermal sensor. The sensor could be inside the processor or in a separate component near the processor. CMI follows these behaviors with regard to the PROCHOT# signal:

1. Based on the PROCHOT# signal going active, CMI generates an SMI# or SCI (depending on SCI_EN).
2. If the PROCHOT_POL bit is set low, when PROCHOT# goes low, the PROCHOT_STS bit will be set. This is an indicator that the thermal threshold has been exceeded. If the PROCHOT_EN bit is set, then when PROCHOT_STS goes active, either an SMI# or SCI# will be generated (depending on the SCI_EN bit being set). The power management software (BIOS or ACPI) can then take measures to start reducing the temperature. Examples include shutting unneeded subsystems, or halting the processor.
3. By setting the PROCHOT_POL bit to high, another SMI# or SCI# can optionally be generated when the PROCHOT# signal goes back high. This allows the software (BIOS or ACPI) to turn off the cooling methods.

Note: PROCHOT# assertion does not cause a TCO event message in S3 or S4. The level of the signal is not reported in the heartbeat message.

27.7.2 Processor Passive Cooling

The FORCE_THTL bit allows the BIOS to force passive cooling independent of the ACPI software (which uses the THTL_EN and THTL_DTY bits). It has the following behavior:

1. If this bit is set, CMI will start throttling using the ratio in the PROCHOT_DTY field.
2. If this bit is turned off, (cleared) CMI will stop throttling, unless the THTL_EN bit is set (indicating that ACPI software is attempting throttling).

If both the THTL_EN and FORCE_THTL bits are both set, then the IICH must use the duty cycle defined by the PROCHOT_DTY field, not the THTL_DTY field. (i.e., PROCHOT_DTY has higher priority).

27.7.3 On-Demand Passive Cooling

This is a method to cool the system by throttling the processor. The mode is initiated by software setting the THTL_EN or THTL_DTY bits.

Behavioral Description:

1. Software sets the THTL_DTY bits to select throttle ratio and the THTL_EN bit to enable the throttling.
2. Throttling results in STOPCLK# active for a minimum time of 12.5% and a maximum of 87.5%. The period is 1024 PCI clocks. Thus, the STOPCLK# signal can be active for as little as 128 PCI clocks or as much as 896 PCI clocks. The actual slowdown (and cooling) of the processor will depend on the instruction stream, because the processor is allowed to finish the current instruction. Furthermore, CMI waits for the STOP-GRANT cycle before starting the count of the time the STOPCLK# signal is active.
3. CMI will perform the Go-C2/Ack-C2 and Go-C0/Ack-C0 messages for throttling, just as if it were making transitions to/from a C2 state.

27.7.4 Active Cooling

Active cooling involves fans. The GPIO signals from CMI can be used to turn on/off a fan.

27.8 Event Input Signals, Messages and Their Usage

CMI has various input signals that trigger specific events. This section describes those signals and how they should be used.

27.8.1 PWRBTN# – Power Button

CMI PWRBTN# signal operates as a “Fixed Power Button” as described in the *ACPI Specification*. PWRBTN# signal has a 16 ms debounce on the input. The state transition descriptions are included in Table 27-35. The transitions start as soon as the PWRBTN# is pressed (but after the debounce logic), and does not depend on when the Power Button is released. A power button override will force a transition to S5, even if PWROK is not active.

Table 27-35. Transitions Due to Power Button

Present State	Event	Transition / Action	Comment
S0/Cx	PWRBTN# goes low	SMI# or SCI generated (depending on SCI_EN)	Software will typically initiate a Sleep state.
S1-S5	PWRBTN# goes low	Wake Event. Transitions to S0 state.	Standard wakeup ***Note:*** Could be impacted by SLP_S4# minimum assertion.
G3	PWRBTN# pressed	None	No effect since no power. Not latched nor detected.
S0 -S4	PWRBTN# held low for at least 4 consecutive seconds	Unconditional transition to S5 state.	No dependence on processor (such as Stop-Grant cycles) or any other subsystem.

27.8.1.1 Power Button Override Function

If PWRBTN# is observed active for at least 4 consecutive seconds, then the state machine must unconditionally transition to the G2/S5 state, regardless of present state (S0-,S4) even if PWROK is not active. In this case, the transition to the G2/S5 state must not depend on any particular response from the processor (such as a Stop-Grant cycle), nor any similar dependency from any other subsystem.

Note: The 4-second PWRBTN# assertion must only be used if a system lock-up has occurred. The 4-second timer starts counting when CMI is in a S0 state. If the PWRBTN# signal is asserted and held active when the system is in a suspend state (S1,S5), the assertion causes a wake event. Once the system has resumed to the S0 state, the 4-second timer starts. The PWRBTN# status is readable to check if the button is currently being pressed or has been released. The status is taken after the debounce, and is readable via the PWRBTN_LVL bit.

Note: During the time that the SLP_S4# signal is stretched for the minimum assertion width (if enabled by D31:F0:A4h bit 3), the Power Button is not a wake event. As a result, it is conceivable that the user will press and continue to hold the Power Button waiting for the system to awake. Since a 4-second press of the Power Button is already defined as an Unconditional Power down, the power button timer will be forced to inactive while the power-cycle timer is in progress. Once the power-cycle timer has expired, the Power Button awakes the system. Once the minimum SLP_S4# power cycle expires, the Power Button must be pressed for another 4 to 5 seconds to create the Override condition to S5.

27.8.1.2 Sleep Button

The *Advanced Configuration and Power Interface (ACPI) Specification, Rev. 2.0b* defines an optional Sleep button. It differs from the power button in that it only is a request to go from S0 to S1-S4 (not S5). Also, in an S5 state, the Power Button can wake the system, but the Sleep Button cannot.

Although CMI does not include a specific signal designated as a Sleep Button, one of the GPIO signals can be used to create a “Control Method” Sleep Button See the *ACPI Specification* for implementation details.

27.8.2 RI# – Ring Indicate Signal

The Ring Indicator can cause a wake event (if enabled) from the S1, S3, S4 or S5 states. Table 27-36 shows when the wake event is generated or ignored in different states. If in the G0/S0/Cx states, CMI generates an interrupt based on RI# active, and the interrupt will be set up as a Break event.

Note: There is no filtering on the RI# signal. Any debounce filtering must be done externally.

Table 27-36. Transitions Due to RI# Signal

Present State	Event	RI_EN	Event
S0	RI# Active	X	Ignored
S1, S3, S4, S5	RI# Active	0 1	Ignored Wake Event

27.8.3 PME# – PCI Power Management Event

The PME# signal comes from a PCI device to request that the system be restarted. The PME# signal can generate an SMI, SCI, or optionally a Wake event. The event occurs when the PME# signal goes from high to low. No event is caused when it goes from low to high.

27.8.4 SYS_RESET# Button

When the SYS_RESET# button is detected as active after the debounce logic (100 ms debounce on the input, same as PWRBTN#), CMI will attempt to perform a “graceful” reset, by waiting up to 25 ms, +/- 2ms for SM Bus to go idle. If SM Bus is idle when the button is detected active, the reset will occur immediately, otherwise the counter will start. If at any point during the count SM Bus goes idle, the counter will be reset and the full system reset will occur. If, however, the counter expires and SM Bus is still active, a full system reset will be forced upon the system even though SMBus activity is still occurring.

Once the reset is asserted, it will remain asserted for approximately 1 ms, regardless of whether the SYS_RESET# input remains asserted or not. It cannot occur again until SYS_RESET# has been detected inactive after the debounce logic, and the system is back to a full S0 state with PLTRST# inactive.

Note: If bit 3 of the CF9h I/O register is set then SYS_RESET# will result in a full power cycle reset.

27.8.5 Processor Thermal Trip

If THRMTRIP# goes active, the processor is indicating an overheat condition, and will immediately transition to an S5 state. However, since the processor has overheated, it will not respond to the STPCLK# pin with a stop grant special cycle. Therefore, CMI will not wait for one. Immediately upon seeing THRMTRIP# low, CMI will initiate a transition to the S5 state, drive signals SLP_S3#, SLP_S4#, SLP_S5# low, and set the CTS bit. The transition will generally look like a power button override.

When a THRMTRIP# event occurs, CMI will power down immediately without following the normal S0 -> S5 path.

If the processor is running extremely hot and is heating up, it is possible (although very unlikely) that components around it, such as the IICH, are no longer executing cycles properly. Therefore, if THRMTRIP# goes active, and IICH is relying on state machine logic to perform the power down, the state machine may not be working, and the system will not power down.

CMI will follow this flow for THRMTRIP#.

1. At boot (PLTRST# low), THRMTRIP# ignored.
2. After power-up (PLTRST# high), if THRMTRIP# sampled active, SLP_S3#, SLP_S4#, and SLP_S5# fire, and normal sequence of sleep machine starts.
3. Until sleep machine enters the S5 state, SLP_S3#, SLP_S4#, and SLP_S5# stay active, even if THRMTRIP# is now inactive. This is the equivalent of “latching” the thermal trip event.
4. When S5 state reached, go to step #1, otherwise stay here. If CMI never gets to S5, CMI does not reboot until power is cycled.

During boot, THRMTRIP# is ignored until SLP_S3#, PWROK, VRMPWRGD, and PLTRST# are all ‘1’. During entry into a powered-down state (due to S3,S4, S5 entry, power cycle reset, etc.) THRMTRIP# is ignored until either SLP_S3# = 0, or PWROK = 0, or VRMPWRGD = 0.

Note: A processor thermal trip event will

1. Set the AFTERG3_EN bit
2. Clear the PWRBTN_STS bit
3. Clear all the GPE0_EN register bits
4. Clear the SMB_WAK_STS bit only if SMB_WAK_STS was set due to SMBus slave receiving message and not set due to SMBAlert.

Note: The THRMTRIP# pin must be glitch free.

27.8.6 SATA SCI

The SATA logic can cause an SCI, but not an SMI or wake event. When the SATA logic causes an SCI, the SATA_SCI_STS bit will be set. The SCI handler enables the SCI and clears the SCI via bits in the SATA unit.

27.8.7 PCI Express* PME Event Message

PCI Express* ports and the IMCH have the ability to cause PME using messages. When a PME messages is receive, CMI will set the PCI-EXP-STS bit.

27.9 Alternate (ALT) Access Mode

Before entering a low power state, several registers from powered down parts may need to be saved. In the majority of cases, this is not an issue, as registers have read and write paths. However, several of the ISA (legacy) compatible registers are either read-only or write-only. To get data out of write-only registers, and to restore data into read-only registers, the IICH implements an alternate access mode.

If the ALT access mode is entered and exited after reading the registers of IICH timer (8254), the timer starts counting faster (13.5 ms). The following steps listed below can cause problems:

1. BIOS enters ALT access mode for reading the IICH timer related registers.
2. BIOS exits ALT access mode.
3. BIOS continues through the execution of other needed steps and passes control to the operating system.

After getting control in step #3, if the operating system does not reprogram the system timer again, the timer ticks may be happening faster than expected. For example DOS and its associated software assume that the system timer is running at 54.6 ms and as a result the time-outs in the software may be happening faster than expected.

Operating systems (e.g., Microsoft Windows* 98, Windows* 2000, and Windows NT*) reprogram the system timer and therefore do not encounter this problem.

For some other loss (e.g., Microsoft MS-DOS*) the BIOS must restore the timer back to 54.6 ms before passing control to the operating system. If the BIOS is entering ALT access mode before entering the suspend state it is not necessary to restore the timer contents after the exit from ALT access mode.

27.9.1 Write Only Registers with Read Paths in Alternate Access Mode

The registers described in Table 27-37 have read paths in alternate access mode. The access number field in the table indicates which register will be returned per access to that port.

Table 27-37. Write-Only Registers with Read Paths in Alternate Access Mode (Sheet 1 of 2)

Restore Data						Restore Data	
I/O Addr	# of Reads	Access	Data	I/O Addr	# of Reads	Access	Data
00h	2	1	DMA Chan 0 base address low byte	40h	7	1	Timer Counter 0 status, bits [5:0]
		2	DMA Chan 0 base address high byte			2	Timer Counter 0 base count low byte
01h	2	1	DMA Chan 0 base count low byte			3	Timer Counter 0 base count high byte
		2	DMA Chan 0 base count high byte			4	Timer Counter 1 base count low byte
02h	2	1	DMA Chan 1 base address low byte			5	Timer Counter 1 base count high byte
		2	DMA Chan 1 base address high byte			6	Timer Counter 2 base count low byte
03h	2	1	DMA Chan 1 base count low byte			7	Timer Counter 2 base count high byte
		2	DMA Chan 1 base count high byte	41h	1		Timer Counter 1 status, bits [5:0]
04h	2	1	DMA Chan 2 base address low byte	42h	1		Timer Counter 2 status, bits [5:0]
		2	DMA Chan 2 base address high byte	70h	1		Bit 7 = NMI Enable, Bits [6:0] = RTC Address
05h	2	1	DMA Chan 2 base count low byte	C4h	2	1	DMA Chan 5 base address low byte
		2	DMA Chan 2 base count high byte			2	DMA Chan 5 base address high byte
06h	2	1	DMA Chan 3 base address low byte	C6h	2	1	DMA Chan 5 base count low byte
		2	DMA Chan 3 base address high byte			2	DMA Chan 5 base count high byte
07h	2	1	DMA Chan 3 base count low byte	C8h	2	1	DMA Chan 6 base address low byte
		2	DMA Chan 3 base count high byte			2	DMA Chan 6 base address high byte
08h	6	1	DMA Chan 0-3 Command[2]	CAh	2	1	DMA Chan 6 base count low byte
		2	DMA Chan 0-3 Request			2	DMA Chan 6 base count high byte
		3	DMA Chan 0 Mode: Bits(1:0) = “00”	CCh	2	1	DMA Chan 7 base address low byte
		4	DMA Chan 1 Mode: Bits(1:0) = “01”			2	DMA Chan 7 base address high byte
		5	DMA Chan 2 Mode: Bits(1:0) = “10”	CEh	2	1	DMA Chan 7 base count low byte
		6	DMA Chan 3 Mode: Bits(1:0) = “11”.			2	DMA Chan 7 base count high byte

Notes:
1. The OCW1 register must be read before entering Alternate Access Mode.
2. Bits 5, 3, 1, and 0 return 0.
3. The additional write-only registers are described in their respective sections.

Table 27-37. Write-Only Registers with Read Paths in Alternate Access Mode (Sheet 2 of 2)

Restore Data				Restore Data			
I/O Addr	# of Reads	Access	Data	I/O Addr	# of Reads	Access	Data
20h	12	1	PIC ICW2 of Master controller	D0h	6	1	DMA Chan 4-7 Command[2]
		2	PIC ICW3 of Master controller			2	DMA Chan 4-7 Request
		3	PIC ICW4 of Master controller			3	DMA Chan 4 Mode: Bits(1:0) = “00”
		4	PIC OCW1 of Master controller[1]			4	DMA Chan 5 Mode: Bits(1:0) = “01”
		5	PIC OCW2 of Master controller			5	DMA Chan 6 Mode: Bits(1:0) = “10”
		6	PIC OCW3 of Master controller			6	DMA Chan 7 Mode: Bits(1:0) = “11”.
		7	PIC ICW2 of Slave controller				
		8	PIC ICW3 of Slave controller				
		9	PIC ICW4 of Slave controller				
		10	PIC OCW1 of Slave controller[1]				
		11	PIC OCW2 of Slave controller				
		12	PIC OCW3 of Slave controller				

Notes:
1. The OCW1 register must be read before entering Alternate Access Mode.
2. Bits 5, 3, 1, and 0 return 0.
3. The additional write-only registers are described in their respective sections.

27.9.2 PIC Reserved Bits

Many bits within the PIC are reserved, and must have certain values written in order for the PIC to operate properly. Therefore, there is no need to return these values in alternate (ALT) access mode. When reading PIC registers from 20h and A0h, the reserved bits shall return the values listed in Table 27-38.

Table 27-38. PIC Reserved Bits Return Values

PIC Reserved Bits	Value Returned
ICW2(2:0)	000
ICW4(7:5)	000
ICW4(3:2)	00
ICW4(0)	0
OCW2(4:3)	00
OCW3(7)	0
OCW3(5)	Reflects bit 6
OCW3(4:3)	01

27.9.3 Read-Only Registers with Write Paths in ALT Access Mode

The registers described in Table 27-39 have write paths to them in ALT access mode. Software will restore these values after returning from a powered down state. These registers must be handled special by software. When in normal mode, writing to the base address/count register also writes to the current address/count register. Therefore, the base address/count must be written first, then the part is put into alternate access mode and the current address/count register is written.

Only bits 3:0 of the DMA Status Registers listed below are writable.

Table 27-39. Register Write Accesses in Alternate Access Mode

I/O Address	Register Write Value
08h	DMA Status Register for channels 0-3.
D0h	DMA Status Register for channels 4-7.

27.10 System Power Supplies, Planes, and Signals

27.10.1 Power Plane Control with SLP_S3#, SLP_S4# and SLP_S5#

The usage of SLP_S3# and SLP_S4# depend on whether the platform is configured for S3-Cold.

- S3-Cold — The SLP_S3# output signal can be used to cut power to the system core supply, since it will only go active for the STR state (typically mapped to ACPI S3). Power must be maintained to system memory, CMI Resume Well, and to any other circuits that need to generate Wake signals from the STR state.

Cutting power to the core may be done via the power supply, or by external FETs to the motherboard.

The SLP_S4# and SLP_S5# output signal can be used to cut power to the system core supply, as well as power to the system memory, since the context of the system is saved on the disk. Cutting power to the memory may be done via the power supply, or by external FETs to the motherboard. The SLP_S4# output signal is used to remove power to additional subsystems that are powered during SLP_S3#.

SLP_S5# output signal can be used to cut power to the system core supply, as well as power to the system memory, since the context of the system is saved on the disk. Cutting power to the memory may be done via the power supply, or by external FETs to the motherboard.

27.10.2 SLP_S4# and Suspend-To-RAM Sequencing

The system memory suspend voltage regulator is controlled by Glue logic. The SLP_S4# signal must be used to remove power to system memory rather than the SLP_S5# signal. The SLP_S4# logic in CMI provides a mechanism to fully cycle the power to the DRAM and/or detect if the power is not cycled for a minimum time.

Note: To utilize the hardware-enforced minimum DRAM power-down feature that is enabled by the SLP_S4# Assertion Stretch Enable bit (Section 27.3.1.3, “Offset A4h: GEN_PMCON_3 - General PM Configuration 3 Register” bit 3), the DRAM power must be controlled by the SLP_S4# signal.

27.10.3 PWROK Signal

The PWROK input must go active based on the core supply voltages becoming valid. PWROK must not go high until at least 99 ms after Vcc3_3 and Vcc1_5 have reached their nominal values.This is required to meet the 100 ms delay from valid power to PLTRST# deassertion in the *PCI Specification, Rev. 2.3*.

1. Traditional designs have an active-low reset button electrically ANDed with the PWROK signal from the power supply and the processors voltage regulator module. If this is done with CMI, the PWROK_FLR bit will be set. CMI treats this internally as if the RSMRST# signal had gone active. However, it is not treated as a full power failure. If PWROK goes inactive and then active (but RSMRST# stays high), then CMI will reboot (regardless of the state of the AFTERG3 bit). If the RSMRST# signal also goes low before PWROK goes high, then this is a full power failure, and the reboot policy is controlled by the AFTERG3 bit.
2. SYSRESET# is recommended for implementing the system reset button. This saves the external logic that is needed when the PWROK input is used. Additionally it allows for better handling of the SM-Bus and processor resets, and avoids improperly reporting or power failures.
3. PWROK and RSMRST# are sampled using the RTC clock. Therefore, low times that are less than one RTC clock period may not be detected by CMI.
4. In the case of true PWROK failure, PWROK will go low first before the VRMPWRGD.
5. If the PWROK input is used to implement the system reset button, CMI does not provide any mechanism to limit the amount of time that the processor is held in reset. The platform must externally guarantee that maximum reset assertion specs are met.

27.10.4 CPUPWRGD Signal

This signal is connected to the processor and is derived from two inputs: VRMPWRGD signal (from the processor's VRM) AND'd with the PWROK signal that comes from the system power supply.

27.10.5 Controlling Leakage and Power Consumption During Low-Power States

To control leakage in the system, various signals will tri-state or go low during some low-power states.

General principles (these are board-level guidelines and are NOT CMI behavioral rules):

- All signals going to powered down planes (either internal or external) must be either tri-states or driven low.
- Signals with pull-up resistors must not be low during low-power states. This is to avoid the power consumed in the pull-up resistor.
- Buses must be halted (and held) in a known state to avoid a floating input (perhaps to some other device). Floating inputs can cause extra power consumption.

Based on the above principles, the following measures are taken:

- During S3 (STR), all signals attached to powered down planes will be tri-stated or driven low.

27.10.6 VRMPWROK

The VRMPWROK signal is generated by the processor's VRM. It indicates that the voltage outputs from the VRM are on and within spec.

27.11 Legacy Power Management Theory of Operation

27.11.1 Overview

Instead of relying on ACPI software, legacy power management uses BIOS and various hardware mechanisms. The scheme relies on the concept of detecting when individual subsystems are idle, detecting when the whole system is idle, and detecting accesses are attempted to idle subsystems.

However, the OS is assumed to at least be APM enabled. Without APM calls, there is no quick way to know when the system is idle between keystrokes. CMI does not support burst modes.

27.11.2 APM Power Management

CMI has a timer, when enabled by the 1MIN_EN bit in the SMI Control and Enable register, generates a periodic SMI# once per minute. There is also an option to have it generate the SMI# once per 32, 16, or 8 seconds.The SMI handler can check for system activity by reading the DEVTRAP_STS register. If none of the system bits are set, the SMI handler can increment a software counter.

If there is activity, the various bits in the DEVTRAP_STS register will be set. Software clears the bits by writing a 1 to the bit position.

The DEVTRAP_STS register allows for monitoring of various internal devices or Super I/O devices (SP, PP, FDC) on LPC or PCI, keyboard controller accesses, or audio functions on PCI or LPC. Other PCI activity can be monitored by checking the PCI Interrupts.

§ §

28.0 IA-32 Core Interface

28.1 IA-32 Core Interface I/O-Mapped Register Details

Table 28-1. IA-32 Core Interface Signal State

Signal Name	S3 Hot	S3 Cold	S5
A20M#	Low	Off	Off
CPUSLP#	Low	Off	Off
IGNNE#	Low	Off	Off
INIT#	Low	Off	Off
INTR	Low	Off	Off
NMI	Low	Off	Off
SMI#	Low	Off	Off
STPCLK#	Low	Off	Off
FERR#	X	Low	Low

Notes:
X = Don't care
ND = Not Determined. May be high or low depending on programming.

Table 28-2. Summary of IA-32 Core Interface Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
61h	61h	"Offset 61h: NMI_STS_CNT - NMI Status and Control Register" on page 1098	00h
70h	70h	"Offset 70h: NMI_EN - NMI Enable (and Real Time Clock Index) Register" on page 1099	80h
92h	92h	"Offset 92h: PORT92 - Fast A20 and Init Register" on page 1100	00h
F0h	F0h	"Offset F0h: COPROC_ERR - Coprocessor Error Register" on page 1100	00h
CF9h	CF9h	"Offset CF9h: RST_CNT - Reset Control Register" on page 1101	00h

28.1.1 Register Descriptions

For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183.

28.1.1.1 Offset 61h: NMI_SC - NMI Status and Control Register

Table 28-3. Offset 61h: NMI_STS_CNT - NMI Status and Control Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 61h **Offset End:** 61h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	SERR_N_NMI_STS	**SERR# NMI Source Status:** 0 = Bit is cleared when bit 2 is set to 1. 1 = Bit is set by any of the sources of the internal SERR on IICH; this includes SERR assertions forwarded from the secondary PCI bus, error from a port, Do_SERR or standard error message from internal Bus 0 functions that generate SERR#. Bit 2 must be cleared in this register in order for this bit to be set. This interrupt source is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 1 and then set it to 0. This bit is read-only. When writing to port 61h, this bit must be 0.		0b	RO
06	IOCHK_NMI_STS	**IOCHK# NMI Source Status:** 0 = Bit is cleared when bit 3 is set to 1. 1 = Bit is set if a legacy agent (via SERIRQ) asserts ISA IOCHK# and bit 3 is cleared (IOCHK_NMI_EN). This interrupt source is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 1and then set bit 3 to 0. When writing to port 61h, this bit must be a 0.		0b	RO
05	TMR2_OUT_STS	**Timer Counter 2 OUT Status:** This bit reflects the current state of the 8254 counter 2 output. Counter 2 must be programmed following any PCI reset for this bit to have a determinate value. When writing to port 61h, this bit must be a 0.		0b	RO
04	REF_TOGGLE	**Refresh Cycle Toggle:** This signal toggles from either 0 to 1 or 1 to 0 at a rate that is equivalent to when refresh cycles would occur. When writing to port 61h, this bit must be a 0. Assumed for compatibility, although no legacy refreshes occur. Must toggle at legacy refresh rate (every 15 µs).		0b	RO
03	IOCHK_NMI_EN	**IOCHK# NMI Enable:** 0 = ISA IOCHK# NMIs are enabled. 1 = ISA IOCHK# NMIs are disabled and cleared.		0b	RW
02	PCI_SERR_EN	**PCI SERR# Enable:** 0 = SERR# NMIs are enabled. 1 = The SERR# NMIs are disabled and cleared.		0b	RW
01	SPKR_DAT_EN	**Speaker Data Enable:** 0 = The SPKR output is a 0. 1 = The SPKR output is equivalent to the Counter 2 OUT signal value.		0b	RW
00	TIM_CNT2_EN	**Timer Counter 2 Enable:** 0 = Counter 2 counting is disabled. 1 = Counting is enabled.		0b	RW

28.1.1.2 Offset 70h: NMI_EN - NMI Enable (and Real Time Clock Index) Register

This register is write-only for normal operation. In Alt-Access mode, this register can be read to find the NMI Enable status and the RTC index value.

Note: The RTC Index field is write-only for normal operation. This field can only be read in Alt-Access Mode. Note, however, that this register is aliased to Port 74h and all bits are readable at that address. See Section 29.2, "RTC I/O Registers" for more information.

Table 28-4. Offset 70h: NMI_EN - NMI Enable (and Real Time Clock Index) Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 70h **Offset End:** 70h	
Size: 8 bit	**Default:** 80h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	NMI_EN	NMI Enable: 0 = NMI sources are enabled. 1 = All NMI sources are disabled.		1b	RWS
06 : 00	RTC_INDX	**Real Time Clock Index (Address):**This data goes to the RTC to select which register or CMOS RAM address is being accessed.		0000000b	RWS

Note: Software must preserve the value of bit 7 at I/O addresses 70h. When writing to this address, software must first read the value, and then write the same value for bit 7 during the sequential address write. Note that port 70h is not directly readable. The only way to read this register is through Alt Access mode. Although RTC Index bits 06:00 are readable from port 74h, bit 7 will always return 0. If the NMI# enable is not changed during normal operation, software can alternatively read this bit once and then retain the value for all subsequent writes to port 70h.

28.1.1.3 Offset 92h: PORT92 - Fast A20 and Init Register

Table 28-5. Offset 92h: PORT92 - Fast A20 and Init Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 92h **Offset End:** 92h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved.		00h	
01	ALT_A20_ GATE	**Alternate A20 Gate:** 0 = The A20M# signal can potentially go active. This bit is ORed with the A20GATE input signal to generate A20M# to the processor. 1 = A20M# signal disabled.		0b	RW
00	INIT_NOW	INIT# forced active: 0 = INIT# is not forced to be active 1 = When this bit transitions from a 0 to a 1, it forces INIT# active for 16 PCI clocks.		0b	RW

28.1.1.4 Offset F0h: COPROC_ERR - Coprocessor Error Register

Table 28-6. Offset F0h: COPROC_ERR - Coprocessor Error Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	OPROC_ ERR	Any value written to this register causes IGNNE# to go active, if FERR# generates an internal IRQ13. In order for FERR# to generate an internal IRQ13, the **Coprocessor Error Enable** bit (Section 17.1.5.5, "Offset 31FFh: OIC - Other Interrupt Control Register", bit 1) must be set to 1.		00h	WO

28.1.1.5 Offset CF9h: RST_CNT - Reset Control Register

Table 28-7. Offset CF9h: RST_CNT - Reset Control Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** CF9h **Offset End:** CF9h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		00h	
03	FULL_RST	**Full Reset:** This bit is used to determine the states of SLP_S3#SLP_S4# and SLP_S5# after a hard reset (not soft reset). 0 = SLP_S3#, SLP_S4# and SLP_S5# are kept high. 1 = Full reset, driving SLP_S3#, SLP_S4# and SLP_S5# low for 3–5 seconds if the following conditions are met: • SYS_RST = 1 (bit 1 of this register) (Hard Reset not soft reset). • RST_CPU is written from 0 to 1(bit 2 of this register). • After PWROK going low (with RSMRST# high), or after two TCO timeouts. When this bit is set, it also causes the full power cycle (SLP_S3/4/5# assertion) in response to SYSRESET#, PWROK#, and Watchdog timer reset sources.		0b	RW
02	RST_CPU	**Reset CPU:** This bit causes either a hard or soft reset to the IA-32 core depending on the state of the SYS_RST bit (bit 1 in this same register). Software causes the reset by setting this bit from a 0 to a 1.		0b	RW
01	SYS_RST	**System Reset:** This bit determines the type of reset caused via RST_CPU (bit 2 of this register). 0 = And RST_CPU goes from 0 to 1 (Soft Reset), then it forces INIT# active for 16 PCI clocks. 1 = And RST_CPU goes from 0 to 1 (Hard Reset), then it forces PLTRST# (and PCIRST#) and SUS_STAT# active for 5 to 6 ms. The IICH main power well is reset when this bit is 1 (and some suspend well logic may also be reset).		0b	RW
00	Reserved	Reserved		0b	

28.2 IA-32 Core Interface Signals

This section provides additional behavioral descriptions of the signals that interface between the IICH and the IA-32 core.

28.2.1 A20M# (Mask A20)

The A20M# signal is active (low) when both of the following conditions are true:

1. The ALT_A20_GATE bit (Bit 1 of PORT92 register) is a '0'.

   and

2. The A20GATE input signal is a '0'.

The A20GATE input signal is expected to be generated by the external microcontroller (KBC).

28.2.2 INIT# (Initialization)

The INIT# signal is active (driven low) based on any one of several events described in Table 28-8. When any of these events occur, INIT# is driven low for 16 PCI clocks, then driven high.

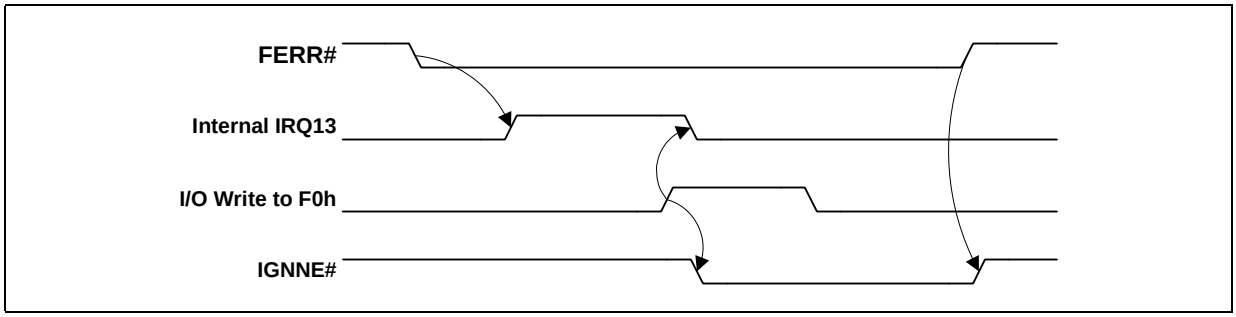
The IICH supports the coprocessor error function with the FERR#/IGNNE# pins. The

Table 28-8. INIT# Going Active

Cause of INIT# Going Active	Comment
Shutdown special cycle from IA-32 core observed on the IICH interconnect (from IMCH).	
PORT92 write, where INIT_NOW (bit 0) transitions from a 0 to a 1.	
PORTCF9 write, where SYS_RST (bit 1) was a 0 and RST_CPU (bit 2) transitions from 0 to 1.	
RCIN# input signal goes low. RCIN# is expected to be driven by the external microcontroller (KBC).	0 to 1 transition on RCIN# must occur before the IICH arms INIT# to be generated again.RCIN# signal is expected to be high during S3-hot and low (due to core power being off) during S3-cold, S4, and S5 states. Transitions on the RCIN# signal in those states (or in the transition to those states) may not necessarily cause the INIT# signal to be generated to the .
CPU BIST	The processor uses two processor pins, INIT# and RESET, to initiate BIST. The processor executes BIST when INIT# is active on RESET's falling edge. Another way to initiate BIST is to enter the RUNBIST command through the TAP serial port. By default, CPU BIST is disabled. In order to enter CPU BIST, software must set the RCBA.CBE bit 2 = '1' and then do a full processor reset using the CF9 register bit 2. NOTE: A3# functions in the same manner that INIT# does as a strap to run BIST.

function is enabled via the COPROC_ERR_EN bit (Device 31, Function 0, Offset D0, Bit 13); refer to Table 28-6 for details. FERR# is tied directly to the Coprocessor Error signal of the . If FERR# is driven active by the , IRQ13 goes active (internally). When it detects a write to the COPROC_ERR register, the IICH negates the internal IRQ13 and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. IGNNE# is never driven active unless FERR# is active.

Figure 28-1. Coprocessor Error Timing Diagram



If COPROC_ERR_EN is not set, then the assertion of FERR# does not generate an internal IRQ13, nor writes to F0h generate IGNNE#.

Non-Maskable Interrupts (NMIs) can be generated by several sources that are described in Table 28-9.

Table 28-9. NMI Sources

Cause of NMI	Comment
SERR# goes active (either internally, externally via SERR# signal, or via message from IMCH)	
ISA IOCHK# goes active via SERIRQ# stream (legacy system Error)	IOCHK# is a legacy signal. CMI does not have this pin, but it may be on the platform.
Watch Dog Timer (LPC bus: logical device 6) first stage 35-bit Down Counter reaches zero.	Enabled by WDT_INT_TYPE field in the WDT Configuration Register.

28.2.3 INTR# (Interrupt Signals)

The behavior of the INTR signal and I/O APIC interrupt signals are described in Chapter 30.0, “Interrupts”.

28.2.4 STPCLK# and CPUSLP# (Stop Clock Request and Processor Sleep Signals)

These active-low signals are controlled by the power management logic. See Chapter 27.0, “Power Management” for more details.

28.2.5 Enhanced Intel SpeedStep Technology (EIST) Signals

Enhanced Intel SpeedStep Technology (EIST) is not supported.

28.2.6 DPSLP# (Deeper Sleep)

DPSLP# is not supported.

29.0 Real Time Clock

29.1 Overview

The Real Time Clock (RTC) module provides a battery backed-up date and time keeping device with two banks of static RAM with 128 bytes each. The first bank has 114 bytes for general purpose usage. Three interrupt features are available: time of day alarm with once a second to once a month range, periodic rates of 122 µs to 500 ms, and end of update cycle notification. Seconds, minutes, hours, days, day of week, month, and year are counted. Daylight savings compensation is optional. The hour is represented in 12 or 24 hour format, and data can be represented in BCD or binary format. The design is meant to be functionally compatible with the Motorola* MS146818B. The time keeping comes from a 32.768 KHz oscillating source, which is divided to achieve an update every second. The lower 14 bytes on the lower RAM block have very specific functions. The first ten are for time and date information. The next four (0Ah to 0Dh) are registers, which configure and report RTC functions.

The time and calendar data must match the data mode (BCD or binary) and hour mode (12 or 24 hour) as selected in register B. The programmer MUST make sure that data stored in these registers is within the reasonable values ranges and represents a possible date and time. The exception to these ranges is to store a value of C0 - FF in the alarm bytes to indicate a “don’t care” situation. All alarm conditions must match to trigger an Alarm Flag, which could trigger an Alarm Interrupt if enabled. The SET bit in register B must be ‘1’ while programming these locations to avoid clashes with update cycles. Access to time and date information is done through the RAM locations. If a RAM read from the ten time and date bytes is attempted during an update cycle, the value read will not necessarily represent the true contents of those locations. Any RAM writes under the same conditions are ignored.

Note: The leap year determination for adding a 29th day to February does not take into account the end-of-the-century exceptions. The logic simply assumes that all years divisible by four are leap years. According to the Royal Observatory Greenwich, years that are divisible by 100 are typically not leap years. In every fourth century (years divisible by 400, like 2000), the 100-year-exception is overridden and a leap-year occurs. Note that the year 2100 is the first time in which the current RTC implementation would incorrectly calculate the leap-year.

29.2 RTC I/O Registers

The RTC internal registers and RAM are organized as two banks of 128 bytes each, called the standard and extended banks. The first 14 bytes of the standard bank contain the RTC time and date information along with four registers, A–D, that are used for configuration of the RTC. The extended bank contains a full 128 bytes of battery backed SRAM, and is accessible even when the RTC module is disabled (via the RTC configuration register). Registers A – D do not physically exist in the RAM.

All data movement between the host CPU and the real-time clock is done through registers mapped to the standard I/O space. The register map appears below in Table 29-1.

Table 29-1. I/O Registers

I/O Locations	If U128E bit = 0	Function:
70h and 74h	Also alias to 72h and 76h	Real-Time Clock (Standard RAM) Index Register ***Note:*** Writes to 72h, 74h, and 76h do not affect NMI enable (bit 7 of 70h)
71h and 75h	Also alias to 73h and 77h	Real-Time Clock (Standard RAM) Target Register
72h and 76h		Extended RAM Index Register (if enabled)
73h and 77h		Extended RAM Target Register (if enabled)

I/O locations 70h and 71h are the standard legacy location for the real-time clock. The map for this bank is shown in Table 29-2. Locations 72h and 73h are for accessing the extended RAM. The extended RAM bank is also accessed using an indexed scheme. I/O address 72h is used as the address pointer and I/O address 73h is used as the data register. Index addresses above 127h are not valid. If the extended RAM is not needed, it may be disabled.

Note: Software must preserve the value of bit 7 at I/O addresses 70h. When writing to this address, software must first read the value, and then write the same value for bit 7 during the sequential address write. Note that port 70h is not directly readable. The only way to read this register is through Alt Access mode. Although RTC Index bits 06:00 are readable from port 74h, bit 7 will always return 0. If the NMI# enable is not changed during normal operation, software can alternatively read this bit once and then retain the value for all subsequent writes to port 70h.

29.3 Real Time Clock Indexed Register Details

The RTC contains two sets of indexed registers that are accessed using the two separate Index and Target registers (70/71h or 72/73h), as shown in Table 29-2.

Table 29-2. RTC (Standard) RAM Bank

Index	Name
00h	Seconds
01h	Seconds Alarm
02h	Minutes
03h	Minutes Alarm
04h	Hours
05h	Hours Alarm
06h	Day of Week
07h	Day of Month
08h	Month
09h	Year
0Ah	Register A
0Bh	Register B
0Ch	Register C
0Dh	Register D
0Eh – 7Fh	114 Bytes of User RAM

Table 29-3. Summary of Real Time Clock Indexed Registers

Offset Start	Offset End	Register ID - Description	Default Value
0Ah	0Ah	"Offset 0Ah: RTC_REGA - Register A (General Configuration)" on page 1107	XXh
0Bh	0Bh	"Offset 0Bh: RTC_REGB - Register B (General Configuration)" on page 1109	X0X00XXXb
0Ch	0Ch	"Offset 0Ch: RTC_REGC - Register C (Flag Register)" on page 1110	00X00000b
0Dh	0Dh	"Offset 0Dh: RTC_REGD - Register D (Flag Register)" on page 1111	10XXXXXXb

29.3.1 Real Time Clock Register Details

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values and are read-only. Writes to reserved locations may cause system failure and unpredictable results.

Note: Reserved bits are read only.

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, "Register Description Tables" on page 183.

29.3.1.1 Offset 0Ah: RTC_REGA - Register A (General Configuration)

Table 29-4. Offset 0Ah: RTC_REGA - Register A (General Configuration) (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:** XXh			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	UIP	**Update in progress:** This bit may be monitored as a status flag. 0 = The update cycle will not start for at least 488 µs. The time, calendar, and alarm information in RAM is always available when the UIP bit is 0. 1 = The update is soon to occur or is in progress.		X	RO

Table 29-4. Offset 0Ah: RTC_REGA - Register A (General Configuration) (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Ah **Offset End:** 0Ah	
Size: 8 bit	**Default:** XXh			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06 : 04	DV	**Division Chain Select:** These three bits control the divider chain for the oscillator, and are not affected by RSMRST# or any other reset signal. DV[2] corresponds to bit 6. **DV2 DV1 DV0 Function** 0 1 0 Normal Operation 1 1 X Divider Reset 0 0 1 Invalid 0 0 0 Invalid		X	RW
03 : 00	RS	**Rate Select:** Selects one of 13 taps of the 15 stage divider chain. The selected tap can generate a periodic interrupt if the PIE bit is set in Register B. Otherwise this tap sets the PF flag of Register C. If the periodic interrupt is not to be used, these bits should all be set to zero. RS3 corresponds to bit 3. RS3 RS2 RS1 RS0 Periodic Rate 0 0 0 0 Interrupt never toggles 0 0 0 1 3.90625 ms 0 0 1 0 7.8125 ms 0 0 1 1 122.070 ms 0 1 0 0 244.141 ms 0 1 0 1 488.281 ms 0 1 1 0 976.5625ms 0 1 1 1 1.953125 ms 1 0 0 0 3.90625 ms 1 0 0 1 7.8125 ms 1 0 1 0 15.625 ms 1 0 1 1 31.25 ms 1 1 0 0 62.5 ms 1 1 0 1 125 ms 1 1 1 0 250 ms 1 1 1 1 500 ms		X	RW

29.3.1.2 Offset 0Bh: RTC_REGB - Register B (General Configuration)

Table 29-5. Offset 0Bh: RTC_REGB - Register B (General Configuration) (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:** X0X00XXXb			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	SET	**Update Cycle Inhibit:** Enables/Inhibits the update cycles. 0 = Update cycle occurs normally once each second. 1 = A current update cycle aborts and subsequent update cycles do not occur until SET is returned to zero. The BIOS may initialize time and calendar bytes safely. ***Note:*** This bit is not affected by RSMRST# nor any other reset signal. ***Note:*** Software must ensure this bit transitions from '1' to '0' once whenever the RTC coin battery is inserted. This is to ensure that the internal RTC time updates occur properly.		X	RW
06	PIE	**Periodic Interrupt Enable:** 0 = Disabled 1 = Allows an interrupt to occur with a time base set with the RS bits of register A. ***Note:*** This bit is cleared by RSMRST#, but not on any other reset.		0h	RW
05	AIE	**Alarm Interrupt Enable:** 0 = Disabled 1 = the Alarm Interrupt Enable (AIE) bit allows an interrupt to occur when the AF is one as set from an alarm match from the update cycle. An alarm can occur once a second, one an hour, once a day, or once a month. ***Note:*** This bit is cleared by RTEST#, but not on any other reset.		X	RW
04	UIE	**Update-ended Interrupt Enable:** 0 = Disabled 1 = Allows an interrupt to occur when the update cycle ends. ***Note:*** This bit is cleared by RSMRST#, but not on any other reset.		0h	RW
03	SQWE	**Square Wave Enable:** The Square Wave Enable bit serves no function in this device, but it is in this register bank to provide compatibility with the Motorola 146818B. There is not a SQW pin on this device. ***Note:*** This bit is cleared by RSMRST#, but not on any other reset.		0h	RW

Table 29-5. Offset 0Bh: RTC_REGB - Register B (General Configuration) (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Bh **Offset End:** 0Bh	
Size: 8 bit	**Default:** X0X00XXXb			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	DM	**Data Mode:** The Data Mode (DM) bit specifies either binary or BCD data representation. 0 = denotes BCD 1 = denotes binary This bit is not affected by RSMRST# nor any other reset signal.		X	RW
01	HOURFORM	**Hour Format:** This bit indicates the hour byte format. 0 = Twelve-hour mode is selected. In twelve hour mode, the seventh bit represents AM as zero and PM as one. 1 = Twenty-four hour mode is selected. This bit is not affected by RSMRST# nor any other reset signal.		X	RW
00	DSE	**Daylight Savings Enable:** 0 = Disabled 1 = Triggers two special hour updates per year when set to one. One is on the first Sunday in April, where time increments from 1:59:59 AM to 3:00:00 AM. The other is the last Sunday in October when the time first reaches 1:59:59 AM, it is changed to 1:00:00 AM. The time must increment normally for at least two update cycles (seconds) previous to these conditions for the time change to occur properly. These special update conditions do not occur when the DSE bit is set to zero. The days for the hour adjustment are those specified in United States federal law as of 1987, which is different than previous years. ***Note:*** This bit is not affected by RSMRST# nor any other reset signal.		X	RW

29.3.1.3 Offset 0Ch: RTC_REGC - Register C (Flag Register)

Table 29-6. Offset 0Ch: RTC_REGC - Register C (Flag Register) (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:** 00X00000b			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	IRQF	**Interrupt Request Flag:** Interrupt Request Flag = (PF * PIE) + (AF * AIE) + (UF *UFE). This also causes the RTC Interrupt to be asserted. ***Note:*** This bit is cleared upon RSMRST# or a read of Register C.		0b	RO

Table 29-6. Offset 0Ch: RTC_REGC - Register C (Flag Register) (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:** 00X00000b			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06	PF	**Periodic Interrupt Flag:** 0 = No taps are specified. 1 = Set when the tap as specified by the RS bits of register A is one. ***Note:*** This bit is cleared upon RSMRST# or a read of Register C.		0b	RO
05	AF	**Alarm Flag:** 0 = All Alarm values do not match the current time 1 = All Alarm values match the current time. ***Note:*** This bit is cleared upon RTEST# or a read of Register C.		X	RO
04	UF	**Update-ended Flag:** 0 = Update cycle not detected 1 = This bit is set immediately following an update cycle for each second. ***Note:*** The bit is cleared upon RSMRST# or a read of Register C.		0b	RO
03 : 00	Reserved	**Reserved**		000b	

29.3.1.4 Offset 0Dh: RTC_REGD - Register D (Flag Register)

Table 29-7. Offset 0Dh: RTC_REGD - Register D (Flag Register)

Description:					
View: IA F	**Base Address:** RTC Standard RAM Bank			**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 10XXXXXXb			**Power Well:** RTC	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	VRT	**Valid RAM and Time Bit:** This bit is hard-wired to '1' in the RTC power well. This bit should always be written as a 0 for write cycle, however it will return a 1 for read cycles.		1b	RW
06	Reserved	**Reserved:** This bit always returns a 0 and should be set to 0 for write cycles.		0b	
05 : 00	DA	**Date Alarm:** These bits store the date of month alarm value. If set to 000000, then a don't care state is assumed. The host must configure the dates alarm for these bits to do anything, yet they can be written at any time. If the date alarm is not enabled, these bits return zeros to mimic the functionality of the Motorola 146818B. These bits are not affected by any reset assertion.		XXXXXXb	RW

29.4 Update Cycles

An update cycle occurs once a second, if the SET bit of register B is not asserted and the divide chain is properly configured. During this procedure, the stored time and date are incremented, overflow is checked, a matching alarm condition is checked, and the time and date are rewritten to the RAM locations. The update cycle will start at least 488 µs after the UIP bit of register A is asserted, and the entire cycle will not take more than 1984 µs to complete. The time and date RAM locations (0–9) is disconnected from the external bus during this time.

To avoid update and data corruption conditions, external RAM access to these locations can safely occur upon the detection of either of two conditions.

1. When an updated-ended interrupt is detected, almost 999 ms is available to read and write the valid time and date data.
2. If the UIP bit of Register A is detected to be low, there is at least 488 µs before the update cycle begins.

Warning: The overflow conditions for leap years and daylight savings adjustments are based on more than one date or time item. To ensure proper operation when adjusting the time, the new time and data values should be set at least two seconds before one of these conditions (leap year, daylight savings time adjustments) occurs.

29.5 Interrupts

The real-time clock interrupt is internally routed within the IICH both to the I/O APIC and the 8259. It is mapped to interrupt vector 8. This interrupt does not leave the IICH prior to connection to the interrupt controller, nor is it shared with any other interrupt. IRQ8# from the SERIRQ stream is ignored. However, the High Performance Event Timers can also be mapped to IRQ8#; in this case, the RTC interrupt is blocked.

29.6 Lockable RAM Ranges

The RTC's battery-backed RAM supports two 8-byte ranges that can be locked via the PCI configuration space. If the locking bit is set, the corresponding range in the RAM is not readable or writeable. A write cycle to those locations has no effect. A read cycle to those locations does not return the locations actual value (resultant value is undefined).

Once a range is locked, the range can be unlocked only by a hard reset, which will invoke the BIOS and allow it to relock the RAM range.

29.7 Century Rollover

The IICH detects a roll over when the Year byte (RTC I/O space, index offset 09h) transitions from 99 to 00 (e.g., a rollover from December 31, 2099, 11:59:59 p.m. to 12:00:00 a.m on January 1st, 2100). Upon detecting the rollover, the IICH sets the NEWCENTURY_STS bit (TCOBASE + 04h, bit 7). If the system is in an S0 state, this causes an SMI#. The SMI# handler can update registers in the RTC RAM that are associated with the century value.

If the system is in a sleep state (S3 and S5) when the century rollover occurs, the IICH also sets the NEWCENTURY_STS bit, but no SMI# is generated. When the system resumes from the sleep state, BIOS should check the NEWCENTURY_STS bit and update the century value in the RTC RAM.

29.8 Month and Year Alarms

This function is not supported.

§ §

30.0 Interrupts

30.1 Overview

Only level-triggered interrupts can be shared. PCI interrupts (PIRQs) are inherently shared on the board. These must, therefore, be programmed as level-triggered.

Table 30-1 and Table 30-2 show the mapping of the various interrupts in Non-APIC and APIC modes. Table 30-3 lists the interrupt signals action in the associated power state

Table 30-1. Interrupt Options - 8259 Mode

IRQ	SERIRQ	Pin	Internal Modules
0	No	No	8254 Counter 0, MMT 0
1	Yes	No	
2	No	No	8259 2 cascade only
3	Yes	No	Option for PIRQx
4	Yes	No	Option for PIRQx
5	Yes	No	Option for PIRQx
6	Yes	No	Option for PIRQx
7	Yes	No	Option for PIRQx
8	No	No	RTC, MMT 1
9	Yes	No	Option for PIRQx, SCI, TCO
10	Yes	No	Option for PIRQx, SCI, TCO
11	Yes	No	Option for PIRQx, SCI, TCO, MMT 2
12	Yes	No	Option for PIRQx
13	No	No	FERR# Logic
14	Yes	Yes	PIRQx, SATA Primary (legacy mode)
15	Yes	Yes	PIRQx, SATA Secondary (legacy mode)

Notes:

1. If an interrupt is used for PCI IRQ[A:H], SCI, or TCO, it must not be used for ISA (legacy)-style interrupts (via SERIRQ).
2. PIRQ[A-D] do not come out on the pins any longer.
3. In 8259 mode, PCI interrupts are mapped to IRQ3, 4, 5, 6, 7, 9, 10, 11, 12, 14, or 15.
4. If IRQ11 is used for MMT 2, software must ensure IRQ11 is not shared with any other devices to guarantee the proper operation of MMT 2. The hardware does not prevent sharing of IRQ11.

Table 30-2. Interrupt Options - APIC Mode

IRQ #	SERIRQ	Pin	PCI Message	Internal Modules
0	No	No	No	Cascade from 8259 1
1	Yes	No	Yes	
2	No	No	No	8254 Counter 0, MMT 0 (legacy mode)
3	Yes	No	Yes	
4	Yes	No	Yes	
5	Yes	No	Yes	
6	Yes	No	Yes	
7	Yes	No	Yes	
8	No	No	No	RTC, MMT 1 (legacy mode)
9	Yes	No	Yes	Option for SCI, TCO
10	Yes	No	Yes	Option for SCI, TCO
11	Yes	No	Yes	Option for SCI, TCO
12	Yes	No	Yes	
13	No	No	No	FERR# Logic
14	Yes	Yes	Yes	SATA Primary (legacy mode)
15	Yes	Yes	Yes	SATA Secondary (legacy mode)
16	PIRQA	No	Yes	See Section 30.5 for how internal devices are mapped.
17	PIRQB	No		
18	PIRQC	No		
19	PIRQD	No		
20	N/A	PIRQE		Option for SCI, TCO, and HPET (High Precision Event Timer) For other internal devices, see Section 30.5
21	N/A	PIRQF		
22	N/A	PIRQG		
23	N/A	PIRQH		
24	N/A	GPIO[16]	N/A	External GPIO driven interrupt
25	N/A	GPIO[17]	N/A	External GPIO driven interrupt
26	N/A	GPIO[20]	N/A	External GPIO driven interrupt
27	N/A	GPIO[21]	N/A	External GPIO driven interrupt
28	N/A	GPIO[23]	N/A	External GPIO driven interrupt
29	N/A	GPIO[24]	N/A	External GPIO driven interrupt
30	N/A	GPIO[28]	N/A	External GPIO driven interrupt
31	N/A	GPIO[30]	N/A	External GPIO driven interrupt
32	N/A	GPIO[31]	N/A	External GPIO driven interrupt

Notes:

1. If an interrupt is used for PCI IRQ[A:H], SCI, or TCO, it must not be used for ISA (legacy)-style interrupts (via SERIRQ).
2. PIRQ[A-D]do not come out on the pins any longer.
3. In APIC mode, the PCI interrupts A:H are mapped to IRQ[16:23].
4. When programming the polarity of internal interrupt sources on the APIC, interrupts 0 through 15 receive active-high internal interrupt sources; interrupts 16 through 23 receive active-low internal interrupt sources.
5. IRQ24-39 are externally driven interrupts through GPIO pins, enabled if GPIO_IRQ_STRAP_STS field of Extended Test Mode Register3 (ETR3) is set to 1 (set by externally pulling strap siu2_txd_ad18 to LOW on the rising edge of PWROK) and if each of the 16 IRQ capable GPIO pins is configured to work in alternative mode (in GPIO_USE_SEL1/GPIO_USE_SEL2 registers).

Table 30-2. Interrupt Options - APIC Mode

33	N/A	GPIO[33]	N/A	External GPIO driven interrupt
34	N/A	GPIO[34]	N/A	External GPIO driven interrupt
35	N/A	GPIO[40]	N/A	External GPIO driven interrupt
36	N/A	GPIO[18]	N/A	External GPIO driven interrupt
37	N/A	GPIO[19]	N/A	External GPIO driven interrupt
38	N/A	GPIO[25]	N/A	External GPIO driven interrupt
39	N/A	GPIO[27]	N/A	External GPIO driven interrupt

Notes:

1. If an interrupt is used for PCI IRQ[A:H], SCI, or TCO, it must not be used for ISA (legacy)-style interrupts (via SERIRQ).
2. PIRQ[A-D]do not come out on the pins any longer.
3. In APIC mode, the PCI interrupts A:H are mapped to IRQ[16:23].
4. When programming the polarity of internal interrupt sources on the APIC, interrupts 0 through 15 receive active-high internal interrupt sources; interrupts 16 through 23 receive active-low internal interrupt sources.
5. IRQ24-39 are externally driven interrupts through GPIO pins, enabled if GPIO_IRQ_STRAP_STS field of Extended Test Mode Register3 (ETR3) is set to 1 (set by externally pulling strap siu2_txd_ad18 to LOW on the rising edge of PWROK) and if each of the 16 IRQ capable GPIO pins is configured to work in alternative mode (in GPIO_USE_SEL1/GPIO_USE_SEL2 registers).

Table 30-3. Signals Associated with Interrupt Logic

Signal Name	S3	S5
SERIRQ	Off	Off
PIRQ[A:H]#	Off	Off

30.2 8259 Interrupt Controllers (PIC)

30.2.1 Overview

The IICH incorporates the functionality of two 8259 interrupt controllers that provide system interrupts for the legacy (ISA) compatible interrupt controller (PIC). These interrupts are: system timer, keyboard controller, serial ports, parallel ports, floppy disk, mouse, and DMA channels. In addition, this interrupt controller can support the PCI based interrupts by mapping the PCI interrupt onto the compatible IRQ interrupt line. Each 8259 core supports eight interrupts, numbered 0–7. Table 30-4 shows how the cores are connected.

Table 30-4. 8259 Core Connection (Sheet 1 of 2)

8259	8259 Input	Typical Interrupt Source	Connected Pin / Function
Master	0	Internal	Internal Timer / Counter 0 output or Multimedia Timer 0
	1	Keyboard	IRQ1 via SERIRQ
	2	Internal	Slave Controller INTR output
	3	Serial Port A	IRQ3 via SERIRQ, PIRQx
	4	Serial Port B	IRQ4 via SERIRQ, PIRQx
	5	Parallel Port / Generic	IRQ5 via SERIRQ, PIRQx
	6	Floppy Disk	IRQ6 via SERIRQ, PIRQx
	7	Parallel Port / Generic	IRQ7 via SERIRQ, PIRQx

Table 30-4. 8259 Core Connection (Sheet 2 of 2)

8259	8259 Input	Typical Interrupt Source	Connected Pin / Function
Slave	0	Real Time Clock	Inverted IRQ8# from internal RTC or Multimedia Timer 1
	1	Generic	IRQ9 via SERIRQ, SCI, or TCO, PIRQx
	2	Generic	IRQ10 via SERIRQ, SCI, or TCO, PIRQx
	3	Generic	IRQ11 via SERIRQ, SCI, or TCO, PIRQx
	4	PS/2 Mouse	IRQ12 via SERIRQ, SCI, or TCO, PIRQx
	5	Internal	State Machine output based on processor FERR# assertion. Can optionally be used for SCI or TCO interrupts if FERR# is not needed.
	6	SATA	SATA Primary (legacy mode), SERIRQ, PIRQx
	7	SATA	SATA Secondary (legacy mode), SATA Secondary (legacy mode), SERIRQ, PIRQx

The IICH cascades the slave controller onto the master controller through master controller interrupt input 2. This means there are only 15 possible interrupts for the IICH PIC.

Interrupts can individually be programmed to be edge or level, except for IRQ0, IRQ2, IRQ8#.

Note: Active-low interrupt sources, such as the PIRQ#s, are internally inverted in the IICH. In the following descriptions of the 8259s, the interrupt levels are in reference to the signals at the internal interface of the 8259s, after the required inversions have occurred. Therefore, the term “high” indicates “active”, which means “low” on an originating PIRQ#.

30.2.2 I/O Registers

The interrupt controller registers are located at 20h and 21h for the master controller (IRQ0–7), and at A0h and A1h for the slave controller (IRQ8–13). These registers have multiple functions depending upon the data written to them. Table 30-5 lists the different register possibilities for each address.

Table 30-5. Summary of 8259 Interrupt Controller (PIC) Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
020h, 0A0h	020h, 0A0h	“ICW1[0-1] - Initialization Command Word 1 Register” on page 1119	0001X0XXb
021h, 0A1h	021h, 0A1h	“ICW2[0-1] - Initialization Command Word 2 Register” on page 1120	XXh
21h	21h	“MICW3 - Master Initialization Command Word 3 Register” on page 1121	04h
A1h	A1h	“SICW3 - Slave Initialization Command Word 3 Register” on page 1121	00h
21h, 0A1h	21h, 0A1h	“ICW4[0-1] - Initialization Command Word 4 Register” on page 1122	01h
021h, 0A1h	021h, 0A1h	“OCW1[0-1]- Operational Control Word 1 (Interrupt Mask) Register” on page 1122	00h
020h, 0A0h	020h, 0A0h	“OCW2[0-1] - Operational Control Word 2 Register” on page 1123	001XXXXXb
020h, 0A0h	020h, 0A0h	“OCW3[0-1] - Operational Control Word 3 Register” on page 1124	001XX10b
4D0h	4D0h	“ELCR1 - Master Edge/Level Control Register” on page 1125	00h
4D1h	4D1h	“ELCR2 - Slave Edge/Level Control Register” on page 1126	00

30.2.2.1 ICW1[0-1] - Initialization Command Word 1 Register

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence, during which the following occurs:

1. The Interrupt Mask register is cleared.
2. IRQ7 input is assigned priority 7.
3. The slave mode address is set to 7.
4. Special Mask Mode is cleared and Status Read is set to IRR.

Once this write occurs, the controller expects writes to ICW2, ICW3, and ICW4 to complete the initialization sequence.

Table 30-6. ICW1[0-1] - Initialization Command Word 1 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 020h, 0A0h **Offset End:** 020h, 0A0h	
Size: 8 bit	**Default:** 0001X0XXb			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	Reserved	Reserved. Must be programmed to zero.		000h	
04	Reserved	Reserved. Must be programmed to one.		1	
03	LTIM	**Edge/Level Bank Select:** Disabled. Replaced by the edge/level triggered control registers (ELCR, D31, F0, 4D0h and D31, F0, 4D1h).		X	WO
02	Reserved	Reserved. Must be programmed to zero.		0h	
01	SNGL	**Single or Cascade:** This bit must be programmed to a 0 to indicate that two controllers are operating in cascade mode.		X	WO
00	IC4	**ICW4 Write Required:** This bit must be programmed to a 1 to indicate that ICW4 needs to be programmed.		X	WO

30.2.2.2 ICW2[0-1] - Initialization Command Word 2 Register

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[07:03] is used by the processor to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA (legacy) ICW2 values are 08h for the master controller and 70h for the slave controller.

Table 30-7. ICW2[0-1] - Initialization Command Word 2 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 021h, 0A1h **Offset End:** 021h, 0A1h	
Size: 8 bit	**Default:** XXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	IVBA	**Interrupt Vector Base Address:** These bits define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input.		Xh	WO
02 : 00	IRL	**Interrupt Request Level:** When writing ICW2, these bits must all be 0. During an interrupt acknowledge cycle, these bits are programmed by the interrupt controller with the interrupt to be serviced. This is combined with bits [07:03] to form the interrupt vector driven onto the data bus during the second INTA# cycle. The code is a three bit binary code: Code Master Interrupt Slave Interrupt 000 IRQ0 IRQ8 001 IRQ1 IRQ9 010 IRQ2 IRQ10 011I RQ3 IRQ11 100 IRQ4 IRQ12 101 IRQ5 IRQ13 110 IRQ6 IRQ14 111 IRQ7 IRQ15		Xh	WO

30.2.2.3 MICW3 - Master Initialization Command Word 3 Register

Table 30-8. MICW3 - Master Initialization Command Word 3 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 21h **Offset End:** 21h	
Size: 8 bit	**Default:** 04h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved. Must be programmed to zero.		0h	
02	CCC	**Cascaded Controller Connection:** This bit must always be programmed to a 1. This bit indicates that the slave controller is cascaded on IRQ2. When IRQ8#–IRQ15 is asserted, it goes through the slave controller's priority resolver. The slave controller's INTR output onto IRQ2. IRQ2 then goes through the master controller's priority solver. If it wins, the INTR signal is asserted to the processor, and the returning interrupt acknowledge returns the interrupt vector for the slave controller.		1	WO
01 : 00	Reserved	Reserved. Must be programmed to zero.		0h	

30.2.2.4 SICW3 - Slave Initialization Command Word 3 Register

Table 30-9. SICW3 - Slave Initialization Command Word 3 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** A1h **Offset End:** A1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved. Must be programmed to zero.		0h	
02 : 00	SIC	**Slave Identification Code:** These bits are compared against the slave identification code broadcast by the master controller from the trailing edge of the first internal INTA# pulse to the trailing edge of the second internal INTA# pulse. These bits must be programmed to 02h to match the code broadcast by the master controller. When 02h is broadcast by the master controller during the INTA# sequence, the slave controller assumes responsibility for broadcasting the interrupt vector.		0h	WO

30.2.2.5 ICW4[0-1] - Initialization Command Word 4 Register

Table 30-10. ICW4[0-1] - Initialization Command Word 4 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 21h, 0A1h **Offset End:** 21h, 0A1h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	Reserved	Must be programmed to zero.		0h	
04	SFNM	**Special Fully Nested Mode:** 0 = Disabled by writing a 0 to this bit. 1 = Special fully nested mode is programmed.		0h	WO
03	BUF	**Buffered Mode:** This bit must be programmed to 0 which is non-buffered mode. ***Note:*** Writing '1' will result in undefined behavior.		0h	WO
02	MSBM	**Master/Slave in Buffered Mode:** Must be programmed to 0.		0h	WO
01	AEOI	**Automatic End of Interrupt:** 0 = This bit must normally be programmed to 0. This is the normal end of interrupt. 1 = Automatic End of Interrupt (AEOI) mode is programmed. AEOI is discussed in Section 30.2.7.2.		0h	WO
00	MM	**Microprocessor Mode:** Must be programmed to 1 to indicate that the controller is operating in an Intel Architecture-based system. ***Note:*** Writing '0' will result in undefined behavior.		1	WO

30.2.2.6 OCW1[0-1] - Operational Control Word 1 (Interrupt Mask) Register

Table 30-11. OCW1[0-1]- Operational Control Word 1 (Interrupt Mask) Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 021h, 0A1h **Offset End:** 021h, 0A1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRM	**Interrupt Request Mask:** 0 = The corresponding IRQ mask bit is cleared and interrupt requests are again accepted by the controller. 1 = The corresponding IRQ line is masked. Masking IRQ2 on the master controller also masks the interrupt requests from the slave controller.		00h	RW

30.2.2.7 OCW2[0-1] - Operational Control Word 2 Register

Following a device reset or ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

Table 30-12. OCW2[0-1] - Operational Control Word 2 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 020h, 0A0h **Offset End:** 020h, 0A0h	
Size: 8 bit	**Default:** 001XXXXXb			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 05	REOI	**Rotate and EOI Codes (R, SL, EO):** These three bits control the Rotate and End of Interrupt modes and combinations of the two. 000 Rotate in Auto EOI Mode (Clear) 001 Non-specific EOI command 010 No Operation 011 *Specific EOI Command 100 Rotate in Auto EOI Mode (Set) 101 Rotate on Non-Specific EOI Command 110 *Set Priority Command 111 *Rotate on Specific EOI Command *L0 - L2 Are Used		001h	WO
04 : 03	OCW2_SEL	**OCW2 Select:** When selecting OCW2, bits 04:03 = “00”		X	WO
02 : 00	INT_LS	**Interrupt Level Select:** L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case. Code Interrupt Level CodeInterrupt Level 000I RQ0/8 100IRQ4/12 001 IRQ1/9 101IRQ5/13 010 IRQ2/10 110IRQ6/14 011 IRQ3/11 111IRQ7/15		X	WO

30.2.2.8 OCW3[0-1] - Operational Control Word 3 Register

Table 30-13. OCW3[0-1] - Operational Control Word 3 Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 020h, 0A0h **Offset End:** 020h, 0A0h	
Size: 8 bit	**Default:** 001XX10b			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved.		0h	
06	SMM	**Special Mask Mode:** 0 = The Special Mask Mode will not be used by an interrupt service routine. 1 = The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits. Bit 5, the ESMM bit, must be set for this bit to have any meaning.		0h	WO
05	ESMM	**Enable Special Mask Mode:** 0 = Disable. The SMM bit becomes a "don't care". 1 = Enable the SMM bit to set or reset the Special Mask Mode.		1	WO
04 : 03	O3S	**OCW3 Select:** When selecting OCW3, bits04:03 = "01".		X	WO
02	PMC	**Poll Mode Command:** 0 = Disable. Poll Command is not issued. 1 = Enable. The next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle. An encoded byte is driven onto the data bus, representing the highest priority level requesting service.		X	WO
01 : 00	RRC	**Register Read Command:** These bits provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1=0, bit 0 does not affect the register read selection. When bit 1=1, bit 0 selects the register status returned following an OCW3 read. If bit 0=0, the IRR is read. If bit 0=1, the ISR is read. Following ICW initialization, the default OCW3 port address read is "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read. 00 No Action 01 No Action 10 Read IRQ Register 11 Read IS Register		10b	WO

30.2.2.9 ELCR1 - Master Edge/Level Control Register

In edge mode, (bit[x] = 0), the interrupt is recognized by a low to high transition. In level mode (bit[x] = 1), the interrupt is recognized by a high level. The cascade channel, IRQ2, the heart beat timer (IRQ0), and the keyboard controller (IRQ1), cannot be put into level mode.

Table 30-14. ELCR1 - Master Edge/Level Control Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 4D0h **Offset End:** 4D0h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	ECL7	Edge Level Control IRQ7: 0 = Edge mode. The interrupt is recognized by a low to high transition. 1 = Level mode. The interrupt is recognized by a high level.		0h	RW
06	ECL6	**Edge Level Control IRQ6:** 0 = Edge mode. The interrupt is recognized by a low to high transition. 1 = Level mode. The interrupt is recognized by a high level.		0h	RW
05	ECL5	**Edge Level Control IRQ5:** 0 = Edge mode. The interrupt is recognized by a low to high transition. 1 = Level mode. The interrupt is recognized by a high level.		0h	RW
04	ECL4	**Edge Level Control IRQ4:** 0 = Edge mode. The interrupt is recognized by a low to high transition. 1 = Level mode. The interrupt is recognized by a high level.		0h	RW
03	ECL3	**Edge Level Control IRQ3:** 0 = Edge mode. The interrupt is recognized by a low to high transition. 1 = Level mode. The interrupt is recognized by a high level.		0h	RW
02 : 00	Reserved	Reserved.		0h	

30.2.2.10 ELCR2 - Slave Edge/Level Control Register

In edge mode, (bit[x] = 0), the interrupt is recognized by a low to high transition. In level mode (bit[x] = 1), the interrupt is recognized by a high level. The real time clock, IRQ8#, and the floating point error interrupt, IRQ13, cannot be programmed for level mode.

Table 30-15. ELCR2 - Slave Edge/Level Control Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 4D1h **Offset End:** 4D1h	
Size: 8 bit	**Default:** 00			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	ECL15	**Edge Level Control IRQ15:** 0 = Edge 1 = Level		0h	RW
06	ECL14	**Edge Level Control IRQ14:** 0 = Edge 1 = Level		0h	RW
05	Reserved	Reserved.		0h	
04	ECL12	**Edge Level Control IRQ12:** 0 = Edge 1 = Level		0h	RW
03	ECL11	**Edge Level Control IRQ11**: 0 = Edge 1 = Level		0h	RW
02	ECL10	**Edge Level Control** IRQ10: 0 = Edge 1 = Level		0h	RW
01	ECL9	**Edge Level Control IRQ9**: 0 = Edge 1 = Level		0h	RW
00	Reserved	Reserved. Must be zero.		0h	

30.2.3 Interrupt Handling

30.2.3.1 Generating Interrupts

The PIC interrupt sequence involves three bits, from the IRR, ISR, and IMR, for each interrupt level. These bits are used to determine the interrupt vector returned and status of any other pending interrupts. These bits are defined in Table 30-16.

Table 30-16. Interrupt Handling

Bits	Name	Description
IRR	Interrupt Request Register	This bit is set on a low to high transition of the interrupt line in edge mode, and by an active high level in level mode. This bit is set whether or not the interrupt is masked. However, a masked interrupt will not generate INTR.
ISR	Interrupt Service Register	This bit is set, and the corresponding IRR bit cleared, when an interrupt acknowledge cycle is seen, and the vector returned is for that interrupt.
IMR	Interrupt Mask Register	Determines whether an interrupt is masked. Masked interrupts do not generate INTR.

30.2.3.2 Acknowledging Interrupts

The processor generates an interrupt acknowledge cycle that is translated into an Interrupt Acknowledge Special Cycle to the IICH. The PIC translates this cycle into two internal INTA# pulses expected by the 8259 cores. The PIC uses the first internal INTA# pulse to freeze the state of the interrupts for priority resolution. On the second INTA# pulse, the master or slave sends the interrupt vector to the processor with the acknowledged interrupt code. This code is based upon bits [07:03] of the corresponding ICW2 register, combined with three bits representing the interrupt within that controller.

Table 30-17. Content of Interrupt Vector Byte

Master, Slave Interrupt	Bits [07:03]	Bits [02:00]
IRQ7,15	ICW2[07:03]	111
IRQ6,14		110
IRQ5,13		101
IRQ4,12		100
IRQ3,11		011
IRQ2,10		010
IRQ1,9		001
IRQ0,8		000

30.2.3.3 Hardware/Software Interrupt Sequence

1. One or more of the Interrupt Request lines (IRQ) are raised high in edge mode, or seen high in level mode, setting the corresponding IRR bit.
2. The PIC sends INTR active (high) to the processor if an asserted interrupt is not masked.
3. The processor acknowledges the INTR and responds with an interrupt acknowledge cycle.

4. Upon observing the acknowledge cycle, it is converted into two cycles that the internal 8259 pair can respond to. Each cycle appears as an interrupt acknowledge pulse on the internal INTA# pin of the cascaded interrupt controllers.
5. Upon receiving the first internally generated INTA# pulse, the highest priority ISR bit is set and the corresponding IRR bit is reset. On the trailing edge of the first pulse, a slave identification code is broadcast internally by the master PIC to the slave PIC. The slave controller determines if it must respond with an interrupt vector during the second INTA# pulse.
6. Upon receiving the second internally generated INTA# pulse, the PIC returns the interrupt vector. If no interrupt request is present, the PIC will return vector 7 from the master controller.
7. This completes the interrupt cycle. In AEOI mode the ISR bit is reset at the end of the second INTA# pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

30.2.4 Initialization Command Words (ICW)

Before operation can begin, each 8259 must be initialized. In the IICH, this is a four byte sequence. The four initialization command words are referred to by their acronyms: ICW1, ICW2, ICW3, and ICW4.

The base address for each 8259 initialization command word is a fixed location in the I/O memory space: 20h for the master controller, and A0h for the slave controller.

30.2.4.1 ICW1

An I/O write to the master or slave controller base address with data bit 4 equal to 1 is interpreted as a write to ICW1. Upon sensing this write, the IICH PIC expects three more byte writes to 21h for the master controller or A1h for the slave controller to complete the ICW sequence.

A write to ICW1 starts the initialization sequence during which the following automatically occur:

1. Following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.
2. The Interrupt Mask Register is cleared.
3. IRQ7 input is assigned priority 7.
4. The slave mode address is set to 7.
5. Special Mask Mode is cleared and Status Read is set to IRR.

30.2.4.2 ICW2

The second write in the sequence, ICW2, is programmed to provide bits [07:03] of the interrupt vector that are released during an interrupt acknowledge. A different base is selected for each interrupt controller.

30.2.4.3 ICW3

The third write in the sequence, ICW3, has a different meaning for each controller.

- For the master controller, ICW3 is used to indicate which IRQ input line is used to cascade the slave controller. Within the IICH, IRQ2 is used. Therefore, bit 2 of ICW3 on the master controller is set to a 1, and the other bits are set to 0's.
- For the slave controller, ICW3 is the slave identification code used during an interrupt acknowledge cycle. On interrupt acknowledge cycles, the master

controller broadcasts a code to the slave controller if the cascaded interrupt won arbitration on the master controller. The slave controller compares this identification code to the value stored in its ICW3, and if it matches, the slave controller assumes responsibility for broadcasting the interrupt vector.

30.2.4.4 ICW4

The final write in the sequence, ICW4, must be programmed by both controllers. At the very least, bit 0 must be set to a 1 to indicate that the controllers are operating in an Intel Architecture-based system.

30.2.5 Operation Command Words (OCW)

These command words reprogram the Interrupt Controller to operate in various interrupt modes.

- OCW1 masks and unmasks interrupt lines.
- OCW2 controls the rotation of interrupt priorities when in rotating priority mode, and controls the EOI function.
- OCW3 sets up ISR/IRR reads, enables/disables the Special Mask Mode SMM, and enables/disables polled interrupt mode.

30.2.6 Modes of Operation

30.2.6.1 Fully Nested Mode

In this mode, interrupt requests are ordered in priority from 0 through 7, with 0 being the highest. When an interrupt is acknowledged, the highest priority request is determined and its vector placed on the bus. Additionally, the ISR for the interrupt is set. This ISR bit remains set until:

- The processor issues an EOI command immediately before returning from the service routine; or
- If in AEOI mode, on the trailing edge of the second INTA#.

While the ISR bit is set, all further interrupts of the same or lower priority are inhibited, while higher levels generate another interrupt. Interrupt priorities can be changed in the rotating priority mode.

30.2.6.2 Special Fully Nested Mode

This mode is used in the case of a system where cascading is used, and the priority has to be conserved within each slave. In this case, the special fully-nested mode is programmed to the master controller. This mode is similar to the fully-nested mode with the following exceptions:

- When an interrupt request from a certain slave is in service, this slave is not locked out from the master's priority logic and further interrupt requests from higher priority interrupts within the slave is recognized by the master and initiates interrupts to the processor. In normal nested mode, a slave is masked out when its request is in service.
- When exiting the Interrupt Service routine, software has to check whether the interrupt serviced was the only one from that slave. This is done by sending a Non-Specific EOI command to the slave and then reading its ISR. If it is 0, a non-specific EOI can also be sent to the master.

30.2.6.3 Automatic Rotation Mode (Equal Priority Devices)

In some applications, there are a number of interrupting devices of equal priority. Automatic rotation mode provides for a sequential eight-way rotation. In this mode, a device receives the lowest priority after being serviced. In the worst case, a device requesting an interrupt has to wait until each of seven other devices are serviced at most once.

There are two ways to accomplish automatic rotation using OCW2; the Rotation on Non-Specific EOI Command (R=1, SL=0, EOI=1) and the rotate in automatic EOI Mode which is set by (R=1, SL=0, EOI=0).

30.2.6.4 Specific Rotation Mode (Specific Priority)

Software can change interrupt priorities by programming the bottom priority. For example, if IRQ5 is programmed as the bottom priority device, then IRQ6 is the highest priority device. The Set Priority Command is issued in OCW2 to accomplish this, where: R=1, SL=1, and LO-L2 is the binary priority level code of the bottom priority device.

In this mode, internal status is updated by software control during OCW2. However, it is independent of the EOI command. Priority changes can be executed during an EOI command by using the Rotate on Specific EOI Command in OCW2 (R=1, SL=1, EOI=1 and LO-L2=IRQ level to receive bottom priority.

30.2.6.5 Poll Mode

Poll Mode can be used to conserve space in the interrupt vector table. Multiple interrupts that can be serviced by one interrupt service routine do not need separate vectors if the service routine uses the poll command. Polled Mode can also be used to expand the number of interrupts. The polling interrupt service routine can call the appropriate service routine, instead of providing the interrupt vectors in the vector table. In this mode, the INTR output is not used and the microprocessor internal Interrupt Enable flip-flop is reset, disabling its interrupt input. Service to devices is achieved by software using a Poll Command.

The Poll command is issued by setting P=1 in OCW3. The PIC treats its next I/O read as an interrupt acknowledge, sets the appropriate ISR bit if there is a request, and reads the priority level. Interrupts are frozen from the OCW3 write to the I/O read. The byte returned during the I/O read will contain a ‘1’ in bit 7 if there is an interrupt, and the binary code of the highest priority level in bits 2:0.

30.2.6.6 Edge and Level Triggered Mode

In ISA (legacy) systems this mode is programmed using bit 3 in ICW1, which sets level or edge for the entire controller. In the IICH, this bit is disabled and a new register for edge and level triggered mode selection, per interrupt input, is included. This is the Edge/Level control Registers ELCR1 and ELCR2.

If an ELCR bit is ‘0’, an interrupt request is recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt. If an ELCR bit is ‘1’, an interrupt request is recognized by a high level on the corresponding IRQ input and there is no need for an edge detection. The interrupt request must be removed before the EOI command is issued to prevent a second interrupt from occurring.

In both the edge and level triggered modes, the IRQ inputs must remain active until after the falling edge of the first internal INTA#. If the IRQ input goes inactive before this time, a default IRQ7 vector is returned.

30.2.7 End of Interrupt (EOI) operations

An EOI can occur in one of two fashions: by a command word write issued to the PIC before returning from a service routine, the EOI command; or automatically when AEOI bit in ICW4 is set to 1.

30.2.7.1 Normal EOI

In Normal EOI, software writes an EOI command before leaving the interrupt service routine to mark the interrupt as completed. There are two forms of EOI commands: Specific and Non-Specific. When a Non-Specific EOI command is issued, the PIC clears the highest ISR bit of those that are set to 1. Non-Specific EOI is the normal mode of operation of the PIC within the IICH, as the interrupt being serviced currently is the interrupt entered with the interrupt acknowledge. When the PIC is operated in modes which preserve the fully nested structure, software can determine which ISR bit to clear by issuing a Specific EOI.

An ISR bit that is masked will not be cleared by a Non-Specific EOI if the PIC is in the Special Mask Mode. An EOI command must be issued for both the master and slave controller.

30.2.7.2 Automatic EOI Mode

In this mode, the PIC automatically performs a Non-Specific EOI operation at the trailing edge of the last interrupt acknowledge pulse. From a system standpoint, this mode must be used only when a nested multi-level interrupt structure is not required within a single PIC. The AEOI mode can only be used in the master controller.

30.2.8 Masking Interrupts

30.2.8.1 Masking on an Individual Interrupt Request

Each interrupt request can be masked individually by the Interrupt Mask Register (IMR). This register is programmed through OCW1. Each bit in the IMR masks one interrupt channel. Masking IRQ2 on the master controller masks all requests for service from the slave controller.

30.2.8.2 Special Mask Mode

Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control. For example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion.

The Special Mask Mode enables all interrupts not masked by a bit set in the Mask Register. Normally, when an interrupt service routine acknowledges an interrupt without issuing an EOI to clear the ISR bit, the interrupt controller inhibits all lower priority requests. In the Special Mask Mode, any interrupts may be selectively enabled by loading the Mask Register with the appropriate pattern. The special Mask Mode is set by OCW3.SSMM and OCW3.SMM set, and cleared when OCW3.SSMM and OCW3.SMM are cleared.

30.2.9 Steering of PCI Interrupts

The IICH can be programmed to allow PIRQA#-PIRQH# to be internally routed to interrupts 3-7, 9-12, 14 or 15, through the PARC, PBRC, PCRC, PDRC, PERC, PFRC, PGRC, and PHRC registers in Section 19.2.3.1, “Offset 60h: PARC: PIRQA Routing Control Register” on page 741. The assignment is programmable through the PIRQx

Route Control registers, located at 60–63h and 68–6Bh in Device 31, Function 0. One or more PIRQx# lines can be routed to the same IRQx input. If interrupt steering is not required, the Route registers can be programmed to disable steering.

The PIRQx# lines are defined as active low, level sensitive to allow multiple interrupts on a platform to share a single line across the connector. When PIRQx# is routed to specified IRQ line, software must change the corresponding ELCR1 or ELCR2 register to level sensitive mode. The IICH internally inverts the PIRQx# line to send an active high level to the PIC. When a PCI interrupt is routed onto the PIC, the selected IRQ can no longer be used by an ISA (legacy) device. Internal sources of the PIRQs, including SCI and TCO interrupts, cause the external PIRQ to be asserted. The IICH receives the PIRQ input, like all of the other external sources, and routes it accordingly.

30.3 Advanced Interrupt Controller: APIC

In addition to the standard ISA (legacy)-compatible PIC described in the previous chapter, the IICH also incorporates the APIC.

30.3.1 Interrupt Handling

The I/O APIC handles interrupts very differently than the 8259. Briefly, these differences are:

- **Method of Interrupt Transmission.** The I/O APIC transmits interrupts through memory writes on the normal data path to the processor, and interrupts are handled without the need for the processor to run an interrupt acknowledge cycle.
- **Interrupt Priority.** The priority of interrupts in the I/O APIC is independent of the interrupt number. For example, interrupt 10 can be given a higher priority than interrupt 3.
- **More Interrupts.** The I/O APIC in the IICH supports a total of 40 interrupts (24 interrupts only, when ETR3.GPIO_IRQ_STRAP_STS is 0).
  - When GPIO_IRQ_STRAP_STS field of Extended Test Mode Register3(ETR3) is 0, MRE field in the APIC_VS register reports 17h (indicating 24 supported IRQs).
  - When ETR3.GPIO_IRQ_STRAP_STS is 1 (Strap pulling “siu2_txd_ad18” to LOW on the rising edge of PWROK), IO-APIC can support additional 16, dedicated, GPIO driven IRQs (IRQ24-39). The function of IRQ capable GPIO pins are set to IRQ mode by default if ETR3.GPIO_IRQ_STRAP_STS is 1, but can be changed to GPIO mode for any or all of those pins by programming the GPIO_USE_SEL registers. If the GPIO_USE_SEL register bits are configured to GPIO mode then those GPIO pins can not generate IRQ interrupt to IO-APIC.
  - Note: GPIO pins can not generate interrupt when ETR3.GPIO_IRQ_STRAP_STS is 0.
- **Multiple Interrupt Controllers.** The I/O APIC architecture allows for multiple I/O APIC devices in the system with their own interrupt vectors.

30.3.2 PCI/PCI Express* Message-Based Interrupts

When external devices through PCI/PCI Express* wish to generate an interrupt, they send the message defined in the *PCI Express* Specification* for generating INTA# - INTD#. These are translated internal assertions/deassertions of INTA# - INTD#.

30.3.2.1 Front Side Bus Interrupt Delivery

The IICH requires that the I/O APIC deliver interrupt messages to the processor in a parallel manner, rather than using the I/O APIC serial scheme.

Delivery of interrupts is completed by the IICH writing to a memory location that is snooped by the processor. The processors snoops the cycle to know which interrupt goes active.

The following sequence is used:

1. When the IICH detects an interrupt event (active edge for edge-triggered mode or a change for level-triggered mode), it sets or resets the internal IRR bit associated with that interrupt.
2. Internally, the IICH requests to use the bus in a way that automatically flushes upstream buffers. This can be internally implemented similar to a DMA device request.
3. The IICH delivers the message by performing a write cycle to the appropriate address with the appropriate data. The address and data formats are described in Section 30.3.2.6, “Interrupt Message Format” on page 1133.

30.3.2.2 Edge-Triggered Operation

In this case, the “Assert Message” is sent when there is an inactive-to-active edge on the interrupt.

30.3.2.3 Level-Triggered Operation

In this case, the “Assert Message” is sent when there is an inactive-to-active edge on the interrupt. If after the EOI the interrupt is still active, then another “Assert Message” is sent to indicate that the interrupt is still active.

30.3.2.4 Registers Associated with Front-Side Bus Interrupt Delivery

Capabilities Indication is the capability to support front-side bus interrupt delivery indicated via ACPI configuration techniques. This involves the BIOS creating a data structure that gets reported to the ACPI configuration software.

30.3.2.5 EOI

The mechanism by which the processor may generate an EOI is PCI Express* EOI message.

The PCI Express* EOI message is used by IA-32 core. It is broadcast to the internal IOxAPIC. The data of the EOI message is the vector. This value is compared with all the vectors inside the IOxAPIC, and any match causes RTE[x].RIRR to be cleared. See Section 17.1.6.1 for a description of the EOI vendor-specific message.

30.3.2.6 Interrupt Message Format

CMI writes the interrupt message internally as a 32-bit memory write cycle. It uses the following formats shown in Table 30-18 and Table 30-19 for the address and data.

Table 30-18. Interrupt Delivery Address Format

Bits	Description
31:20	Is always FEEh
19:12	**Destination ID (DID):** This is the same as bits 63:56 of the I/O Redirection Table entry for the interrupt associated with this message.
11:04	**Extended Destination ID** (**EDID):** This is the same as bits 55:48 of the I/O Redirection Table entry for the interrupt associated with this message.
03	**Redirection Hint (DLM):** This bit is used by the processor host bridge to allow the interrupt message to be redirected. 0 = The message is delivered to the agent (processor) listed in bits 19:12. 1 = The message is delivered to an agent with a lower interrupt priority This can be derived from bits 10:8 in the Data Field (see below). The Redirection Hint bit is a 1 if bits 10:8 in the delivery mode field associated with corresponding interrupt are encoded as 001 (Lowest Priority). Otherwise, the Redirection Hint bit is 0
02	**Destination Mode (DSM):** This bit is used only when the Redirection Hint bit is set to 1. If the Redirection Hint bit and the Destination Mode bit are both set to 1, then the logical destination mode is used, and the redirection is limited only to those processors that are part of the logical group as based on the logical ID.
01:00	Will always be 00b

Table 30-19. Interrupt Delivery Data Format

Bits	Description
31:16	Will always be 0000h
15	**Trigger Mode:** 1 = Level,0 = Edge. Same as the corresponding bit in the I/O Redirection Table for that interrupt.
14	**Delivery Status:** 0 = Deassert 1 = Assert If using edge-triggered interrupts, then the bit will always be 1, since only the assertion is sent. If using level-triggered interrupts, then this bit indicates the state of the interrupt input.
13:12	Will always be 0h
11	**Destination Mode:** 1= Logical. 0 = Physical. Same as the corresponding bit in the I/O redirection table
10:08	**Delivery Mode:** This is the same as the corresponding bits in the I/O Redirection Table for that interrupt. 000 = Fixed 100 = NMI 001 = Lowest Priority 101 = INIT 010 = SMI/PMI 110 = Reserved 011 = Reserved 111 = ExtINT
07:00	**Vector:** This is the same as the corresponding bits in the I/O Redirection Table for that interrupt.

30.3.3 APIC Memory-Mapped Register Details

The APIC is accessed via an indirect addressing scheme. Two registers are visible by software for manipulation of most of the APIC registers. These registers are mapped into memory space. The registers are shown inTable 30-20.

Table 30-20. Summary of APIC Registers Mapped in Memory Space“

Offset Start	Offset End	Register ID - Description	Default Value
0000h (4B)	0000h (4B)	“APIC_IDX - Index Register” on page 1135	00h
0010h (4B)	0010h (4B)	“APIC_DAT – Data Register” on page 1136	00h
0040h (4B)	0040h (4B)	“APIC_EOI - EOI Register” on page 1136	00h

30.3.3.1 APIC_IDX - Index Register

The Index register selects which APIC indirect register to be manipulated by software. The selector values for the indirect registers are listed in Table 30-21. Software programs this register to select the desired APIC internal register.

Table 30-21. APIC_IDX - Index Register

Description:					
View: IA F	**Base Address:** FEC00000h			**Offset Start:** 0000h (4B) **Offset End:** 0000h (4B)	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	APIC_Index	This is an 8-bit pointer into the I/O APIC indirect register table listed in Section 30-20, “Summary of APIC Registers Mapped in Memory Space“”.		00h	RW

30.3.3.2 APIC_DAT – Data Register

This register specifies the data to be read or written to the register pointed to by the Index register. This register can be accessed only in Dword quantities. The register is described in Section 30.3.4.

Table 30-22. APIC_DAT – Data Register

Description:					
View: IA F	Base Address: FEC00000h			Offset Start: 0010h (4B) Offset End: 0010h (4B)	
Size: 32 bit	Default: 00h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	WDW	This is a register for the data to be read or written to the APIC indirect register pointed to by the Index register (Table 30-21).		00h	RW

30.3.3.3 APIC_EOI - EOI Register

When a write is issued to this register, the IOxAPIC checks the lower 8 bits written to this register, and compares it with the vector field for each entry in the I/O Redirection Table. When a match is found, RTE.RIRR for that entry is cleared. If multiple entries have the same vector, each of those entries has RTE.RIRR cleared. Only bits 07:00 are used and bits 31:08 are ignored.

Table 30-23. APIC_EOI - EOI Register

Description:					
View: IA F	Base Address: FEC00000h			Offset Start: 0040h (4B) Offset End: 0040h (4B)	
Size: 32 bit	Default: 00h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 08	Reserved	Reserved. software must always write a value of 0 to these bits.		00h	
07 : 00	REC	**Redirection Entry Clear:** When a write is issued to this register, the I/O APIC checks this field, and compares it with the vector field for each entry in the I/O Redirection Table. When a match is found, the Remote_IRR bit for that I/O Redirection Entry is cleared.		00h	WO

30.3.4 Index Registers

Table 30-24 lists the registers which can be accessed within the APIC via the Index (IDX) register. When accessing these registers, accesses must be done a DWord at a time, otherwise unspecified behavior results. Software should not attempt to write to reserved registers. Some reserved registers may return non-zero values when read. For example, software must never access byte 2 from the Data Register before accessing bytes 0 and 1. The hardware does not attempt to recover from a bad programming model in this case.

Table 30-24. APIC Index Register Space

Offset	Symbol	Register
00	APIC_ID	Identification
01	APIC_VS	Version
02 - 0F	-	Reserved
10 - 11h	APIC_RTE[0]	Redirection Table 0
12 - 13h	APIC_RTE[1]	Redirection Table 1
...	...	...
5E - 5Fh	APIC_RTE[39]	Redirection Table 39
60 - FFh	-	Reserved

Note: The supported message delivery type is parallel, i.e., interrupt message from the IOxAPIC is delivered on the (parallel) FSB bus only. Serial APIC bus is not supported.

Table 30-25. Summary of APIC Indexed Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h (4B)	00h (4B)	"APIC_ID – Identification Register" on page 1138	0000h
01h (4B)	01h (4B)	"APIC_VS - Version Register" on page 1138	00170020h
10h at 02h (4B)	11h at 02h (4B)	"APIC_RTE[0-39] - Redirection Table Entry" on page 1139	XXXX0000000 1XXXXh

30.3.4.1 APIC_ID – Identification Register

The APIC ID serves as a physical name of the APIC. The APIC bus arbitration ID for the APIC is derived from its I/O APIC ID. This register is reset to 0 on power-up reset.

Table 30-26. APIC_ID – Identification Register

Description:					
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX			**Offset Start:** 00h (4B) **Offset End:** 00h (4B)	
Size: 32 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 28	Reserved	Reserved		0h	
27 : 24	AID	**APIC Identification:** Software must program this value before using the APIC.		0h	RW
23 : 16	Reserved	Reserved		0h	
15	Scratchpad	Scratchpad		0h	RW
14	Reserved	Reserved.		0h	
13 : 00	Reserved	Reserved		0h	

30.3.4.2 APIC_VS - Version Register

Each I/O APIC contains a hardwired Version Register that identifies different implementation of APIC and their versions. The maximum redirection entry information also is in this register, to let software know how many interrupt are supported by this APIC.

Table 30-27. APIC_VS - Version Register (Sheet 1 of 2)

Description:					
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX			**Offset Start:** 01h (4B) **Offset End:** 01h (4B)	
Size: 32 bit	**Default:** 00170020h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	Reserved	Reserved		0h	
23 :16	MRE	**Maximum Redirection Entries:** This is the entry number (0 being the lowest entry) of the highest entry in the redirection table. This field reports either 17h or 27h depends on the value of GPIO_IRQ_STRAP_STS field of Extended Test Mode Register3 register. • 17h (to indicate 24 interrupts), when ETR3.GPIO_IRQ_STRAP_STS = 0 • 27h (to indicate 40 interrupts), when ETR3.GPIO_IRQ_STRAP_STS = 1 (siu2_txd_ad18 strap configuration)		17h	RO

Table 30-27. APIC_VS - Version Register (Sheet 2 of 2)

Description:					
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX			**Offset Start:** 01h (4B) **Offset End:** 01h (4B)	
Size: 32 bit	**Default:** 00170020h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PRQ	**Pin Assertion Register Supported:** Indicate that the IOxAPIC does not implement the Pin Assertion Register.		0h	RO
14 : 08	Reserved	Reserved		0h	
07 : 00	VS	**Version:** Identifies the implementation version as IOxAPIC. This field reports 20h in all IO-APIC modes (independent of number of enabled IRQ vectors).		20h	RO

30.3.4.3 APIC_RTE[0-39] - Redirection Table Entry

The Redirection Table has a dedicated entry for each interrupt input pin. The information in the Redirection Table is used to translate the interrupt manifestation on the corresponding interrupt pin into an APIC message.

The APIC responds to an edge triggered interrupt as long as the interrupt is held until after the acknowledge cycle has begun. Once the interrupt is detected, a delivery status bit internally to the I/O APIC is set. The state machine steps ahead and wait for an acknowledgment from the APIC unit that the interrupt message was sent. Only then will the I/O APIC be able to recognize a new edge on that interrupt pin. That new edge only results in a new invocation of the handler if its acceptance by the destination APIC causes the Interrupt Request Register bit to go from 0 to 1. (In other words, if the interrupt was not already pending at the destination.) All bits are undefined except for bits 47:17 = 0 and bit 16 = 1.

Table 30-28. APIC_RTE[0-39] - Redirection Table Entry (Sheet 1 of 3)

Description:	Offset: vector 0: 10h-11h, vector 1: 12h-13h, vector 23: 3Eh-3Fh, vector 39: 5Eh-5Fh; vector N: (10h+ (N*2 in Hex)) -(11h + (N*2 in Hex))				
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX	**Vector 0**		**Offset Start:** 10h at 02h (4B) **Offset End:** 11h at 02h (4B)	
Size: 64 bit[a]	**Default:** XXXX00000001XXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 : 56	DID	**Destination ID:** Destination ID of the local APIC.		X	RW
55 : 48	EDID	**Extended Destination ID:** Extended destination ID of the local APIC.		X	RW
47 : 17	Reserved	Reserved		0h	

Table 30-28. APIC_RTE[0-39] - Redirection Table Entry (Sheet 2 of 3)

Description:	Offset: vector 0: 10h-11h, vector 1: 12h-13h, vector 23: 3Eh-3Fh, vector 39: 5Eh-5Fh; vector N: (10h+ (N*2 in Hex)) -(11h + (N*2 in Hex))				
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX	**Vector 0**		**Offset Start:** 10h at 02h (4B) **Offset End:** 11h at 02h (4B)	
Size: 64 bit[a]	**Default:** XXXX00000001XXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
16	MSK	**Mask:** 0 = Not masked; an edge or level on this interrupt pin results in the delivery of the interrupt to the destination. 1 = Masked; interrupts are not delivered nor held pending. Setting this bit after the interrupt is accepted by a local APIC has no effect on that interrupt. This behavior is identical to the device withdrawing the interrupt before it is posted to the processor. It is software's responsibility to deal with the case where the mask bit is set after the interrupt message has been accepted by a local APIC unit but before the interrupt is dispensed to the processor.		1	RW
15	TM	**Trigger Mode:** This field indicates the type of signal on the interrupt pin that triggers an interrupt. 0 = Edge triggered 1 = Level triggered		X	RW
14	RIRR	**Remote IRR:** This bit is used for level triggered interrupts; its meaning is undefined for edge triggered interrupts. 0 = Reset when an EOI message matches the VCT field. 1 = Set when IOxAPIC sends the level interrupt message to the processor.		X	RW
13	POL	**Polarity:** This bit specifies the polarity of each interrupt input. 0 = Active high 1 = Active low		X	RW
12	DS	**Delivery Status:** This field contains the current status of the delivery of this interrupt. Writes to this bit have no effect. 0 = Idle. No activity for this interrupt. 1 = Pending. Interrupt has been injected but delivery is not complete. **For edge triggered interrupts**, this bit indicates that an event has occurred but an interrupt message has yet to be delivered to its targeted destination. Once the interrupt message is delivered, this bit will be cleared. **For level triggered interrupts**, this bit is set when the input event has occurred. This bit is cleared when the interrupt input event is removed. Note that as long as the interrupt input event is active, this bit remains active regardless of whether this interrupt has been delivered or not.		X	RO

Table 30-28. APIC_RTE[0-39] - Redirection Table Entry (Sheet 3 of 3)

Description:	Offset: vector 0: 10h-11h, vector 1: 12h-13h, vector 23: 3Eh-3Fh, vector 39: 5Eh-5Fh; vector N: (10h+ (N*2 in Hex)) -(11h + (N*2 in Hex))				
View: IA I	**Win:Idx:** APIC_WDW:APIC_IDX	**Vector 0**		**Offset Start:** 10h at 02h (4B) **Offset End:** 11h at 02h (4B)	
Size: 64 bit[a]	**Default:** XXXX00000001XXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11	DSM	**Destination Mode:** This field determines the interpretation of the Destination field. 0 = Physical. Destination APIC ID is identified by bits 59:56. 1 = Logical. Destinations are identified by matching bit 63:56 with the Logical Destination in the Destination Format Register and Logical Destination Register in each Local APIC.		X	RW
10 : 08	DLM	**Delivery Mode:** This field specifies how the APICs listed in the destination field must act upon reception of this signal. Delivery Modes will only operate as intended when used in conjunction with a specific trigger mode. The encodings are: 000 Fixed: Deliver the signal on the INTR signal of all processor cores listed in the destination. Trigger Mode can be edge or level. 001 Lowest Priority: Deliver the signal on the INTR signal of the processor core that is executing at the lowest priority among all the processors listed in the specified destination. Trigger Mode can be edge or level. 010 SMI: Not supported. Requires the interrupt to be programmed as edge triggered. 011 Rreserved 100 NMI: Not supported. 101 INIT: Not supported. 110 Reserved 111 ExtINT: Deliver the signal to the INTR signal of all processor cores listed in the destination as an interrupt that originated in an externally connected 8259A compatible interrupt controller. The INTA cycle that corresponds to this ExtINT delivery is routed to the external controller that is expected to supply the vector. Requires the interrupt to be programmed as edge triggered.		X	RW
07 : 00	VCT	**Vector:** This field contains the interrupt vector for this interrupt. Values range between 10h and FEh.		X	RW

a. 64 bit each, accessed as two 32 bit quantities

30.4 PCI Interrupts via /PCI Express*

When external devices through PCI Express* wish to generate an interrupt, they send the message defined in the *PCI Express* Specification* for generating INTA# - INTD#. These are translated into internal assertions/deassertions of INTA# - INTD#.

see Section 17.1.5, “Interrupt Configuration Registers” on page 701

30.5 Serial Interrupt

30.5.1 Overview

The IICH interrupt controller supports a serial IRQ scheme. This allows a single signal to be used to report interrupt requests. The signal used to transmit this information is shared between the interrupt controller and all peripherals that support serial interrupts. The signal line, SERIRQ, is synchronous to the PCI clock, and follows the sustained tri-state protocol that is used by legacy PCI signals. This means that if a device has driven SERIRQ low, it first drives it high synchronous to the PCI clock and releases it after the following PCI clock. The serial IRQ protocol defines this sustained tri-state signaling in the following fashion:

- **S - Sample Phase:** Signal driven low.
- **R - Recovery Phase:** Signal driven high.
- **T - Turn-around Phase:** Signal released.

The IICH supports a message for 21 serial interrupts. These represent the 15 ISA (legacy) interrupts (IRQ0-1, 3-15), the four PCI interrupts, and the control signals SMI# and ISA (legacy) IOCHK#. The serial IRQ protocol does not support the additional APIC interrupts (20–39). The serial interrupt information is transferred using three types of frames:

- **Start Frame:** SERIRQ line driven low by the interrupt controller to indicate the start of IRQ transmission.
- **Data Frames:** IRQ information transmitted by peripherals. The interrupt controller supports 21 data frames.
- **Stop Frame:** SERIRQ line driven low by the interrupt controller to indicate end of transmission and next mode of operation.

30.5.2 Start Frame

The serial IRQ protocol has two modes of operation which affect the start frame:

- **Continuous Mode:** The interrupt controller is solely responsible for generating the start frame.
- **Quiet Mode:** Peripheral initiates the start frame, and the interrupt controller completes it.

The mode that must first be entered when enabling the serial IRQ protocol is continuous mode. In this mode, the IICH asserts the start frame. This start frame is four, six, or eight PCI clocks wide based upon the Serial IRQ Control Register(SCNT.SFPW) field, bits 01:00 at 64h in Device 31, Function 0 configuration space. This is a polling mode.

When the serial IRQ stream enters quiet mode (signaled in the Stop Frame), the SERIRQ line remains inactive and pulled up between the Stop and Start Frame until a peripheral drives the SERIRQ signal low. The IICH senses the line low and continues to drive it low for the remainder of the Start Frame. Since the first PCI clock of the start

frame was driven by the peripheral in this mode, the IICH drives the SERIRQ line low for one PCI clock less than in continuous mode. This mode of operation allows for a quiet, and therefore lower power, operation.

30.5.3 Data Frames

Once the Start frame has been initiated, the SERIRQ peripherals start counting frames based on the rising edge of SERIRQ. Each of the IRQ/DATA frames has exactly three phases of one clock each. The three phases are:

- **Sample Phase:** During this phase, the SERIRQ device drives SERIRQ low if the corresponding interrupt signal is low. If the corresponding interrupt is high, then the SERIRQ devices tri-state the SERIRQ signal. The SERIRQ line remains high due to pull-up resistors (there is no internal pull-up resistor on this signal, an external pull-up resistor is required). A low level during the IRQ0–1 and IRQ2–15 frames indicates that an active-high ISA (legacy) interrupt is not being requested, but a low level during the PCI INT[A:D], SMI#, and ISA (legacy) IOCHK# frame indicates that an active-low interrupt is being requested.
- **Recovery Phase:** During this phase, the device drives the SERIRQ line high if in the Sample Phase it was driven low. If it was not driven in the sample phase, it is tri-stated in this phase.
- **Turn-around Phase:** The device tri-states SERIRQ.

30.5.4 Stop Frame

After all the data frames, a Stop Frame is driven by the IICH. The SERIRQ signal is driven low for two or three PCI clocks. The number of clocks is determined by the SERIRQ configuration register (SCNT.MD field in D31, F0 configuration space). The number of clocks determines the next mode as shown in Table 30-29.

Table 30-29. Stop Frame Definition

Stop Frame Width	Next Mode
2 PCI clocks	**Quite Mode:** Any SERIRQ device initiates a Start Frame
3 PCI clocks	**Continuous Mode:** Only the IICH may initiate a Start Frame

30.5.5 Serial Interrupts Not Supported via SERIRQ

There are three interrupts seen through the serial stream that are not supported by the IICH. These interrupts are generated internally, and are not sharable with other devices within the system. These interrupts are:

- IRQ0:Heartbeat interrupt generated off of the internal 8254 counter 0.
- IRQ8#:RTC interrupt can only be generated internally.
- IRQ13:.....Floating point error interrupt generated off of the processor assertion of FERR#.

The IICH ignores the state of these interrupts in the serial stream, and does not adjust their level based on the level seen in the serial stream.

30.5.6 Special Notes on IRQ14 and IRQ15

IRQ14 and IRQ15 are special interrupts, used by the SATA controller when it is not running in Native IDE mode. If in a legacy mode, IRQ14 and IRQ15 are not accepted from the serial stream, and instead come from these controllers. If the controllers are in Native mode, these interrupts are used by the interrupt controller.

30.5.7 Data Frame Format

Table 30-30 shows the format of the data frames for the associated interrupts. For the legacy (PCI) interrupts (A–D), the output from the IICH is ANDed with the legacy (PCI) input signal. This way, the interrupt can be signaled via both the legacy (PCI) interrupt input signal and via the SERIRQ signal (they are shared).

Table 30-30. Data Frame Format

Data Frame #	Interrupt	Clocks Past Start Frame	Comment
1	IRQ0	2	Ignored. IRQ0 can only be generated via the internal 8524.
2	IRQ1	5	Before port 60h latch
3	SMI#	8	Causes SMI# if low. Sets bit 15 in the SMI_STS register.
4	IRQ3	11	
5	IRQ4	14	
6	IRQ5	17	
7	IRQ6	20	
8	IRQ7	23	
9	IRQ8	26	Ignored. IRQ8# can only be generated internally.
10	IRQ9	29	
11	IRQ10	32	
12	IRQ11	35	
13	IRQ12	38	Before port 60h latch
14	IRQ13	41	Ignored. IRQ13 can only be generated from FERR#.
15	IRQ14	44	Not attached to PATA or SATA logic
16	IRQ15	47	Not attached to PATA or SATA logic
17	IOCHCK#	50	Same as ISA (legacy) IOCHCK# going active.
18	PCI INTA#	53	Drive PIRQA#
19	PCI INTB#	56	Drive PIRQB#
20	PCI INTC#	59	Drive PIRQC#
21	PCI INTD#	62	Drive PIRQD#

§ §

31.0 8254 Timers

31.1 Overview

The IICH contains three counters which have fixed uses. All registers and functions associated with the 8254 timers are in the core power well. The 8254 unit is clocked by a 14.31818 MHz clock.

There is one signal associated with the 8254. It is used to drive the PC speaker.

Table 31-1. SPKR Signal

Signal Name	S3	S5
SPKR	Off	Off

31.2 8254 Timer I/O-Mapped Register Details

Note: For more information on the format of the register description tables that follow in this chapter, see Section 7.1.1, “Register Description Tables” on page 183.

Table 31-2. Summary of 8254 Timer Registers Mapped in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
43h	43h	“Offset 43h: TCW - Timer Control Word Register” on page 1146	XXh
40h at 01h	40h at 01h	“Offset 40h: TSB[0-2] - Interval Timer Status Byte Format Register” on page 1147	0XXXXXXXb
40h at 01h	40h at 01h	“Offset 40h: TCAP[0-2] - Interval Timer Counter Access Ports Register” on page 1148	XXh

31.2.1 Timer Registers

31.2.1.1 Offset 43h: TCW - Timer Control Word Register

This register is programmed prior to any counter being accessed to specify counter modes. Following reset, the control words for each register are undefined and each counter output is 0. Each timer must be programmed to bring it into a known state.

Table 31-3. Offset 43h: TCW - Timer Control Word Register

Description:					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 43h **Offset End:** 43h	
Size: 8 bit	**Default:** XXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :06	CNTSEL	**Counter Select:** The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back command is selected when bits[07:06] are both one. 00 Counter 0 select 01 Counter 1 select 10 Counter 2 select 11 Read Back command		XXb	WO
05 :04	RWMD	**Read/Write Mode Selection:** These bits are the read/write control bits. The actual counter programming is done through the counter port (40h for counter 0, 41h for counter 1, and 42h for counter 2) 00 Counter Latch Command 01 Read/Write Least Significant Byte (LSB) 10 Read/Write Most Significant Byte (MSB) 11 Read/Write LSB then MSB		XXb	RWS
03 :01	CNTMD	**Counter Mode Selection:** These bits select one of six possible modes of operation for the selected counter. 000 0 Out signal on end of count (=0) 001 1 Hardware retriggerable one-shot x10 2 Rate generator (divide by n counter) x11 3 Square wave output 100 4 Software triggered strobe 101 5 Hardware triggered strobe		XXXb	WO
00	BCDCNT	**Binary/BCD Countdown Select:** 0 = Binary countdown is used. The largest possible binary count is 2^{16}. 1 = Binary coded decimal (BCD) count is used. The largest possible BCD count is 10^4.		Xb	WO

There are two special commands that can be issued to the counters through this register, the Read Back command (Section 31.5.3) and the Counter Latch command (Section 31.5.2). When these commands are chosen, several bits within this register are redefined. These register formats are described in Section 31.5.

31.2.1.2 Offset 40h: TSB[0-2] - Interval Timer Status Byte Format Register

Each counter's status byte can be read following a Read Back command. If latch status is chosen (bit 4=0, Read Back command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register (40h for counter 0, 41h for counter 1, and 42h for counter 2) returns the status byte. The status byte returns the values shown in Table 31-4.

Table 31-4. Offset 40h: TSB[0-2] - Interval Timer Status Byte Format Register

Description: These I/O ports can also function as TCAP (see Section 31.2.1.3) based on the settings of TCW (see Section 31.2.1.1).					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 40h at 01h **Offset End:** 40h at 01h	
Size: 8 bit	**Default:** 0XXXXXXXb			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7	COPS	**Counter OUT Pin State:** 0 = The OUT pin of the counter is a 0 1= The OUT pin of the counter is a 1		0b	RO
6	CRS	**Count Register Status:** This bit indicates when the last count written to the Count Register (CR) has been loaded into the counting element (CE). The exact time this happens depends on the counter mode, but until the count is loaded into the counting element (CE), the count value is incorrect. 0 = Count has been transferred from CR to CE and is available for reading. 1 = Null Count. Count has not been transferred from CR to CE and is not yet available for reading.		Xb	RO
05 :04	RWSS	**Read/Write Selection Status:** These reflect the read/write selection made through bits[05:04] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection. 00 Counter Latch Command 01 Read/Write Least Significant Byte (LSB) 10 Read/Write Most Significant Byte (MSB) 11 Read/Write LSB then MSB		XXb	RO
03 :01	MSS	**Mode Selection Status:** These bits return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above. 000 0 Out signal on end of count (=0) 001 1 Hardware retriggerable one-shot x10 2 Rate generator (divide by n counter) x11 3 Square wave output 100 4 Software triggered strobe 101 5 Hardware triggered strobe		XXXb	RO
00	CTS	**Countdown Type Status:** This bit reflects the current countdown type; ether 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.		Xb	RO

31.2.1.3 Offset 40h: TCAP[0-2] - Interval Timer Counter Access Ports Register

Table 31-5. Offset 40h: TCAP[0-2] - Interval Timer Counter Access Ports Register

Description: These I/O ports can also function as TSB (see Section 31.2.1.2) based on the settings of TCW (see Section 31.2.1.1).					
View: IA F	**Base Address:** 0000h (IO)			**Offset Start:** 40h at 01h **Offset End:** 40h at 01h	
Size: 8 bit	**Default:** XXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	CNTP	**Counter Port:** Each counter port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register at port 43h. The counter port is also used to read the current count from the Count Register, and return the status of the counter programming following a Read Back command.		XXh	RW

31.3 Counters

31.3.1 Counter 0, System Timer

This counter functions as the system timer by controlling the state of IRQ0 and is typically programmed for Mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts IRQ0 and decrements the count value by two each counter period. The counter negates IRQ0 when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts IRQ0 when the count value reaches 0, reloads the initial count value and repeats the cycle, alternately asserting and negating IRQ0.

31.3.2 Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for Mode 2 operation and only impacts the period of the REF_TOGGLE bit in Port 61. The initial count value is loaded one counter period after being written to the counter I/O address. The REF_TOGGLE bit has square wave behavior (alternate between 0 and 1) and toggles at a rate based on the value in the counter. Programming the counter to anything other than Mode 2 results in undefined behavior for the REF_TOGGLE bit.

31.3.3 Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h.

31.3.4 Counter Operating Modes

Table 31-6 lists the six operating modes for the interval counters.

Table 31-6. Counter Operating Modes

Mode	Function	Description
0	Out signal on end of count (=0)	Output is '0'. When count goes to 0, output goes to '1' and stays at '1' until counter is reprogrammed.
1	Hardware retriggerable one-shot	Output is '0'. When count goes to 0, output goes to '1' for one clock time.
2	Rate generator (divide by n counter)	Output is '1'. Output goes to '0' for one clock time, then back to '1' and counter is reloaded.
3	Square wave output	Output is '1'. Output goes to '0' when counter rolls over, and counter is reloaded. Output goes to '1' when counter rolls over, and counter is reloaded, etc.
4	Software triggered strobe	Output is '1'. Output goes to '0' when count expires for one clock time.
5	Hardware triggered strobe	Output is '1'. Output goes to '0' when count expires for one clock time.

31.4 Timer Programming

The counter/timers are programmed as follows:

1. Write a control word to select a counter.
2. Write an initial count for that counter.
3. Load the least and/or most significant bytes (as required by Control Word bits 5, 4) of the 16-bit counter.
4. Repeat with other counters.

Only two conventions need to be observed when programming the counters. First, for each counter, the control word must be written before the initial count is written. Second, the initial count must follow the count format specified in the control word (least significant byte only, most significant byte only, or least significant byte and then most significant byte).

A new initial count may be written to a counter at any time without affecting the counter's programmed mode. Counting is affected as described in the mode definitions. The new count must follow the programmed count format.

If a counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same counter. Otherwise, the counter is loaded with an incorrect count.

The Control Word Register at port 43h controls the operation of all three counters. Several commands are available:

- **Control Word Command:** Specifies which counter to read or write, the operating mode, and the count format (binary or BCD).
- **Counter Latch Command:** Latches the current count so that it can be read by the system. The countdown process continues.
- **Read Back Command:** Reads the count value, programmed mode, the current state of the OUT pins, and the state of the Null Count Flag of the selected counter.

31.5 Reading from the Interval Timer

It is often desirable to read the value of a counter without disturbing the count in progress. There are three methods for reading the counters: a Simple Read operation, Counter Latch command, and the Read-Back command. Each is explained in the following sections.

With the Simple Read and Counter Latch command methods, the count must be read according to the programmed format; specifically, if the counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read sequentially. Read, write, or programming operations for other counters may be inserted between them.

31.5.1 Simple Read

The first method is to perform a simple read operation. The counter is selected through port 40h (counter 0), 41h (counter 1), or 42h (counter 2).

Note: Performing a direct read from the counter does not return a determinate value, because the counting process is asynchronous to read operations. However, in the case of counter 2, the count can be stopped by writing to the GATE bit in port 61h.

31.5.2 Counter Latch Command

The Counter Latch Command (Table 31-7), written to port 43h, latches the count of a specific counter at the time the command is received. This command is used to ensure that the count read from the counter is accurate, particularly when reading a two-byte count. The count value is then read from each counter's Count Register through the Counter Port's Access Ports Register (40h for counter 0, 41h for counter 1, and 42h for counter 2).

The count is held in the latch until it is read or the counter is reprogrammed. The count is then unlatched. This allows reading the contents of the counters on the fly without affecting counting in progress. Multiple Counter Latch commands may be used to latch more than one counter. Counter Latch commands do not affect the programmed mode of the counter in any way.

If a counter is latched and then, some time later, latched again before the count is read, the second Counter Latch command is ignored. The count read is the count at the time the first Counter Latch command was issued.

Table 31-7. Counter Latch Command

Bits	Description
07:06	**Counter Selection:** These bits select the counter for latching. 00 Counter 0 01 Counter 1 10 Counter 2 11 The write is interpreted as a read back command.
05:04	**Counter Latch Command:** Write '00' to select the Counter Latch Command.
03:00	Reserved. Must be 0.

31.5.3 Read Back Command

The Read Back command, written to port 43h, latches the count value, programmed mode, and current states of the OUT pin and Null Count flag of the selected counter or counters. The value of the counter and its status may then be read by I/O access to the counter address.

The Read Back command may be used to latch multiple counter outputs at one time. This single command is functionally equivalent to several counter latch commands, one for each counter latched. Each counter's latched count is held until it is read or reprogrammed. Once read, a counter is unlatched. The other counters remain latched until they are read. If multiple count Read Back commands are issued to the same counter without reading the count, all but the first are ignored. Both count and status of the selected counters may be latched simultaneously by setting both bit 5 and bit 4 to 0. If both are latched, the first read operation from that counter returns the latched status. The next one or two reads, depending on whether the counter is programmed for one or two byte counts, returns the latched count. Subsequent reads return an unlatched count.

The Read Back command may additionally be used to latch status information of selected counters. The status of a counter is accessed by a read from that counter's I/O port address. If multiple counter status latch operations are performed without reading the status, all but the first are ignored.

Both count and status of the selected counters may be latched simultaneously in any or all of the counters by selecting the counter during the register write. This is functionally the same as issuing two consecutive, separate Read Back commands. If multiple count and/or status Read Back commands are issued to the same counters without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter will return the latched status, regardless of which was latched first. The next one or two reads, depending on whether the counter is programmed for one or two type counts, return the latched count. Subsequent reads return unlatched count.

Table 31-8. Read Back Command

Bits	Description
07:06	**Read Back Command:** Must be '11' to select the Read Back Command.
05	**Latch Count of Selected Counters:** 0 = Current count value of the selected counters are latched 1 = Current count value of the selected counters are not latched
04	**Latch Status of Selected Counters:** 0 = Status of the selected counters are latched 1 = Status of the selected counters are not latched
03	**Counter 2 Select:** 0 = Counter 2 count and/or status are not latched 1 = Counter 2 count and/or status are latched
02	**Counter 1 Select:** 0 = Counter 1 count and/or status are latched 1 = Counter 1 count and/or status are not latched
01	**Counter 0 Select:** 0 = Counter 0 count and/or status are latched. 1 = Counter 0 count and/or status are not latched
00	Reserved. Must be 0.

§ §

32.0 High Precision Event Timers

Note: This section documents the CMI-specific behavior and the generic *HPET (High Precision Event Timers) Specification, Revision 1.0a.*

32.1 Overview

This function provides a set of timers that can be used by the operating system. The timers are defined such that in the future, the OS may be able to assign specific timers to be used directly by specific applications. Each timer can be configured to cause a separate interrupt. This specification allows for a block of 32 timers, with support for up to eight blocks, for a total of 256 timers. The timers are implemented as a single counter with a set of comparators. The counter increases monotonically. Each timer includes a value register and a comparator. Each individual timer can generate an interrupt when the value in its value register matches value in the main counter. Some of the timers can be enabled to generate a periodic interrupt.

The registers associated with these timers are mapped to a memory space (much like the I/O APIC). However, it is not implemented as a standard PCI function. The BIOS reports to the operating system the location of the register space. The hardware can support an assignable decode space; However, the BIOS sets this space prior to handing it over to the OS. It is not expected that the OS move the location of these timers once they are set by the BIOS.

Note: For additional information see Section 1.3, “Referenced Documents and Related Websites”.

32.2 Register Details

The timer registers are memory mapped in a non-indexed scheme. This allows the CPU to directly access each register without having to use an index register. The timer register space is 1024 bytes. The registers are generally aligned on 64-bit boundaries.

General Behavioral Rules:

- Software must not attempt to read or write across register boundaries. For example, a 32-bit access must be to offset x0h, x4h, x8h, or xCh.
- 32-bit accesses must not be to 01h, 02h, 03h, 05h, 06h, 07h, 09h, 0Ah, 0Bh, 0Dh, 0Eh, or 0Fh. Any accesses to these offsets results in an unexpected behavior and may result in a master abort. However, these accesses may not result in system hangs.
- 64-bit accesses can only be to x0h and must not cross 64-bit boundaries.
- Software must not write to read-only registers.
- Software must not expect any particular or consistent value when reading reserved registers or bits.
- The timer register space is memory mapped to a 1 K block.

- There are four possible memory address ranges beginning at:

1. FED0_0000h
2. FED0_1000h
3. FED0_2000h
4. FED0_3000h

- The choice of address ranges will be selected by configuration bits in the High Performance Timer Configuration Register (in the memory-mapped chipset configuration area) Table 17-26, “Offset 3404h: HPTC - High Performance Precision Timer Configuration Register” on page 705.
- All registers are implemented in the Core well, and all bits are reset by PLTRST#.
- Reads to reserved registers or bits return a value of 0.

Note: Reads to reserved registers or bits returns a value of 0.

Note: Software must not attempt to lock the memory-mapped I/O ranges for High-Precision Timers. If attempted, the lock is not honored, which means potential deadlock conditions may occur.

Table 32-1. Summary of HPET Registers Mapped in Memory Space

Offset Start	Offset End	Register ID - Description	Default Value
000h	007h	“Offset 000h: GCAP_ID - General Capabilities and ID Register” on page 1155	0429B17F8086A201h
010h	017h	“Offset 010h: GEN_CONF - General Configuration Register” on page 1156	0000000000000000h
020h	027h	“Offset 020h: GINTR_STA - General Interrupt Status Register” on page 1157	0000000000000000h
0F0h	0F7h	“Offset 0F0h: MAIN_CNT - Main Counter Value Register” on page 1158	Xh
100h at 20h	107h at 20h	“Offset 100h: HPTCC[0-2] - Timer n Configuration and Capabilities Register” on page 1159	Xh
108h at 20h	10Fh at 20h	“Offset 108h: HPTCV[0-2] - Timer n Comparator Value Register” on page 1163	Xh

32.2.1 Register Descriptions

32.2.1.1 Offset 000h: GCAP_ID - General Capabilities and ID Register

General Behavioral Rules:

- Writes to this register must not be attempted by software.
- Software can read the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 00h or 04h, but not to offsets 01h, 02h, 03h, 05h, 06h, or 07h.
- 64-bit accesses can only be done to 00h.

Table 32-2. Offset 000h: GCAP_ID - General Capabilities and ID Register

Description:					
View: IA F	**Base Address:** HPTC			**Offset Start:** 000h **Offset End:** 007h	
Size: 64 bit	**Default:** 0429B17F8086A201h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :32	COUNTER_CLK_PER_CAP	**Main Counter Tick Period:** This read-only field indicates the period at which the counter increments in femptoseconds (10^-15 seconds). This returns 0429B17Fh when read indicating a period of 69841279 fs (69.841279 ns).		0429B17Fh	RO
31 :16	VENDOR_ID_CAP	**Vendor ID Capability:** These bits return 8086h when read. This is a 16-bit value assigned to Intel.		8086h	RO
15	LEG_RT_CAP	**Legacy Replacement Rout Capable:** This bit is always one when read, as the Legacy Replacement Interrupt Rout is supported.		1h	RO
14	Reserved	**Reserved:** This bit returns zero when read.		0h	
13	COUNT_SIZE_CAP	**Counter Size Capability:** Indicates that the main counter is 64 bits wide. This bit returns one when read.		1h	RO
12 :08	NUM_TIM_CAP	**Number of Timer Capability:** This field indicates the number of timers in this block. This value in this field is 02h = Three timers.		02h	RO
07 :00	REV_ID	**Revision Identification:** This field indicates which revision of the specification is implemented. The value in this field is 01h.		01h	RO

32.2.1.2 Offset 010h: GEN_CONF - General Configuration Register

General Behavioral Rules:

- Software can access the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 010h or 014h, but not to offsets 011h, 012h, 013h, 015h, 016h, or 017h.
- 64-bit accesses can only be done to 010h.

Table 32-3. Offset 010h: GEN_CONF - General Configuration Register

Description:					
View: IA F	**Base Address:** HPTC			**Offset Start:** 010h **Offset End:** 017h	
Size: 64 bit	**Default:** 0000000000000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :02	Reserved	**Reserved:**		000000000000h	RO
01	LEG_RT_CNF	**Legacy Replacement Route:** If the ENABLE_CNF and LEG_RT_CNF bits are set, then the interrupts are routed as follows: Timer 0 is routed to IRQ0 in 8259 or IRQ2 in the I/O APIC Timer 1 is routed to IRQ8 in 8259 or IRQ8 in the I/O APIC Timer 2-n is routed as per the routing in the timer *n* config registers. 0 = If the LEG_RT_CNF bit is not set, the individual routing bits for each of the timers are used. 1 = Legacy Rout: If the LEG_RT_CNF bit is set, the individual routing bits for timers 0 and 1 (APIC or FSB) have no impact.		0b	RW
00	ENABLE_CNF	**Overall Enable:** 0 = The main counter halts (does not increment) and no interrupts are caused by any of these timers. 1 = Enable any of the timers to generate interrupts. For level-triggered interrupts, if an interrupt is pending when the ENABLE_CNF bit is changed from 1 to 0, the interrupt status indications (in the various Txx_INT_STS bits) are not cleared. Software must write to the Txx_INT_STS bits to clear the interrupts.		0b	RW

32.2.1.3 Offset 020h: GINTR_STA - General Interrupt Status Register

General Behavioral Rules:

- Software can access the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 020h or 024h, but not to offsets 021h, 022h, 023h, 025h, 026h, or 027h.
- 64-bit accesses can only be done to 020h.

Table 32-4. Offset 020h: GINTR_STA - General Interrupt Status Register

Description:					
View: IA F	**Base Address:** HPTC			**Offset Start:** 020h **Offset End:** 027h	
Size: 64 bit	**Default:** 0000000000000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :03	Reserved	**Reserved**		00h	RO
02	T02_INT_STS	**Timer 2 Interrupt Active:** Same functionality as Timer 0.		0h	RW
01	T01_INT_STS	**Timer 1 Interrupt Active:** Same functionality as Timer 0.		0h	RW
00	T00_INT_STS	**Timer 0 Interrupt Active:** The functionality of this bit depends on whether the edge or level-triggered mode is used for this timer: If set to level-triggered mode: This bit is set by hardware if the corresponding timer interrupt is active. Once the bit is set, it can be cleared by software writing a 1 to the same bit position. Writes of 0 to this bit have no effect. If set to edge-triggered mode: This bit must be ignored by software. Software must always write 0 to this bit.		0h	RWC

32.2.1.4 Offset 0F0h: MAIN_CNT - Main Counter Value Register

General Behavioral Rules:

- Software can access the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 0F0h or 0F4h.
- 32-bit accesses must not be done starting at: 0F1h, 0F2h, 0F3h, 0F5h, 0F6h, or 0F7h.
- 64-bit accesses can be done to 0F0h.
- Writes to this register must only be done while the counter is halted.
- Reads to this register return the current value of the main counter.
- 32-bit counters will always return zero for the upper 32 bits of this register.
- If 32-bit software attempts to read a 64-bit counter, it must first halt the counter. Since this will delay the interrupts for all of the timers, this must be done only if the consequences are understood. It is strongly recommended that 32-bit software only operate the timer in 32-bit mode.

Table 32-5. Offset 0F0h: MAIN_CNT - Main Counter Value Register

Description:					
View: IA F	**Base Address:** HPTC			**Offset Start:** 0F0h **Offset End:** 0F7h	
Size: 64 bit	**Default:** Xh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :00	COUNTER_VAL_63_0	**Counter Value:** Bits 63:00 of the counter. ***Notes:*** 1. Writes to this register must only be done while the counter is halted. 2. Reads to this register return the current value of the main counter. 3. 32-bit counters always return zero for the upper 32 bits of this register. 4. If 32-bit software attempts to read a 64-bit counter, it must first halt the counter. Since this delays the interrupts for all of the timers, this must be done only if the consequences are understood. It is strongly recommended that 32-bit software only operates the timer in 32-bit mode. 5. Reads to this register are monotonic. No two consecutive reads return the same value. The second of two reads always returns a larger value, unless the timer has rolled over to 0.		Xh	RW

32.2.1.5 Offset 100h: HPTCC[0-2] - Timer *n* Configuration and Capabilities Register

General Behavioral Rules:

- Software can access the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 1x0h or 1x4h. 64-bit accesses can be done to 1x0h.
- 32-bit accesses must not be done to 1x1h, 1x2h, 1x3h, 1x5h, 1x6h, 1x7h.

Note: The letter *n* can be 0, 1 or 2, referring to Timer 0, 1 or 2.

Table 32-6. Offset 100h: HPTCC[0-2] - Timer n Configuration and Capabilities Register (Sheet 1 of 3)

Description: Timer 0: 100 – 107h, Timer 1: 120 – 127h, Timer 2: 140 – 147h, Timer *n*: (20h * *n*) +100h - (20h * *n*) + 107h					
View: IA F	**Base Address:** HPTC			**Offset Start:** 100h at 20h **Offset End:** 107h at 20h	
Size: 64 bit	**Default:** Xh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :56	Reserved	**Reserved:** These bits return 0 when read.		0h	RO
55 :52	TIMERn_INT_ROUT_CAP	**Timer Interrupt Route Capability:** Timer 0, 1:Bits 52, 53, 54, and 55 in this field (corresponding to IRQ 20, 21, 22, and 23) have a value of 1. Writes will have no effect. Timer 2:Bits 43, 52, 53, 54, and 55 in this field (corresponding to IRQ 11, 20, 21, 22, and 23) have a value of 1. Writes will have no effect. If IRQ 11 is used for High Precision Event Timer #2, software must ensure IRQ 11 is not shared with any other devices to guarantee the proper operation of High Precision Event Timer #2.		X	RO
51 :44	Reserved	**Reserved:** These bits return 0 when read.		0h	RO
43	TIMERn_INT_ROUT_CAP	**Timer Interrupt Route Capability:** Timer 0, 1:Bits 52, 53, 54, and 55 in this field (corresponding to IRQ 20, 21, 22, and 23) have a value of 1. Writes will have no effect. Timer 2:Bits 43, 52, 53, 54, and 55 in this field (corresponding to IRQ 11, 20, 21, 22, and 23) have a value of 1. Writes will have no effect. If IRQ 11 is used for High Precision Event Timer #2, software must ensure IRQ 11 is not shared with any other devices to guarantee the proper operation of High Precision Event Timer #2.		X	RO
42 :14	Reserved	**Reserved:** These bits return 0 when read.		0h	RO
Note: Reads or writes to unimplemented timers must not be attempted. Reads from any unimplemented registers return an undetermined value.					

Table 32-6. Offset 100h: HPTCC[0-2] - Timer n Configuration and Capabilities Register (Sheet 2 of 3)

Description: Timer 0: 100 – 107h, Timer 1: 120 – 127h, Timer 2: 140 – 147h, Timer *n*: (20h * *n*) +100h - (20h * *n*) + 107h					
View: IA F	**Base Address:** HPTC			**Offset Start:** 100h at 20h **Offset End:** 107h at 20h	
Size: 64 bit	**Default:** Xh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
13 :09	TIMERn_INT_ROUT_CNF	**Interrupt Route:** (where *n* is the timer number: 00 to 31). This 5-bit field indicates the routing for the interrupt to the I/O APIC. A maximum value of 32 interrupts is supported. The default is 00h. Software writes to this field to select which interrupt in the I/O (x)APIC used for this timer's interrupt. If the value is not supported by this particular timer, then the value read back does not match what is written. The software must only write valid values. ***Notes:*** 1. If the Legacy Replacement Rout bit is set, then Timers 0 and 1 have a different routing, and this bit field has no effect for those two timers. 2. Timer 0,1: The software is responsible to make sure it programs a valid value (decimal 20, 21, 22, or 23) for this field. The logic does not check the validity of the value written. 3. Timer 2: The software is responsible to make sure it programs a valid value (decimal 11, 20, 21, 22, or 23) for this field. The logic does not check the validity of the value written.		Xh	RW
08	TIMERn_32MODE_CNF	0 = **Timer *n* 32-bit Mode:** (where *n* is the timer number: 00 to 31). Software can set this bit to force a 64-bit timer to behave as a 32-bit timer. Timer 0: Bit is read/write and defaults to 0.64 bit 1 = 32 bit Timers 1 and 2: Hardwired to 0. Writes have no effect (since these two timers are 32 bits).		Xh	RW or RO
07	Reserved	**Reserved:**		0b	RO
06	TIMERn_VAL_SET_CNF	**Timer n Value Set:** Software uses this bit only for timers that have been set to periodic mode. 0 = Disabled. Software does NOT have to write this bit back to 0 (it automatically clears). 1 = By writing this bit to a 1, the software is allowed to directly set the timer's accumulator. ***Note:*** Software must not write a 1 to this bit position if the timer is set to non-periodic mode. ***Note:*** This bit returns zero when read. Writes will only have an effect for Timer 0 if it is set to periodic mode. Writes have no effect for Timers 1 and 2.		Xh	RW or RO
05	TIMERn_SIZE_CAP	**Timer *n* Size:** (where *n* is the timer number: 00 to 31). This read-only field indicates the size of the timer. 0 = 32 bits 1 = 64 bits Timer 0: Value is 1 (64 bits). Timers 1 and 2: Value is 0 (32 bits).		Xh	RO
04	TIMERn_PER_INT_CAP	**Periodic Interrupt Capable:** (where *n* is the timer number: 00 to 31). If this read-only bit is 1, then the hardware supports a periodic mode for this timer's interrupt. Timer 0: Hardwired to 1 (supports the periodic interrupt). Timers 1 and 2: Hardwired to 0 (does not support periodic interrupts), so the bit is always read as zero.		Xh	RO
Note: Reads or writes to unimplemented timers must not be attempted. Reads from any unimplemented registers return an undetermined value.					

Table 32-6. Offset 100h: HPTCC[0-2] - Timer n Configuration and Capabilities Register (Sheet 3 of 3)

Description: Timer 0: 100 – 107h, Timer 1: 120 – 127h, Timer 2: 140 – 147h, Timer *n*: (20h * *n*) +100h - (20h * *n*) + 107h					
View: IA F	**Base Address:** HPTC			**Offset Start:** 100h at 20h **Offset End:** 107h at 20h	
Size: 64 bit	**Default:** Xh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	TIMERn_TYPE_CNF	**Timer n Type:** (where *n* is the timer number: 00 to 31). Timer 0:Bit is read/write. 0 = Disable timer to generate a periodic interrupt. 1 = Enable timer to generate a periodic interrupt. Timers 1, 2: Hardwired to 0. so bit access is Read Only.		Xh	RW or RO
02	TIMERn_INT_ENB_CNF	**Timer n Interrupt Enable:** (where *n* is the timer number: 00 to 31). This bit must be set to enable timer n to cause an interrupt when it times out. 0 = Disable. The timer still counts but does not cause an interrupt. 1 = Enable.		0h	RW
01	TIMERn_INT_TYPE_CNF	**Timer Interrupt Type:** (where *n* is the timer number: 00 to 31) 0 = The timer interrupt is edge triggered. This means that an edge-type interrupt is generated. If another interrupt occurs, another edge is generated. 1 = The timer interrupt is level triggered. This means that a level-triggered interrupt is generated. The interrupt is held active until it is cleared by writing to the bit in the General Interrupt Status Register. If another interrupt occurs before the interrupt is cleared, the interrupt remains active.		0h	RW
00	Reserved	**Reserved:** This bit returns zero when read.		0h	RO
Note: Reads or writes to unimplemented timers must not be attempted. Reads from any unimplemented registers return an undetermined value.					

32.2.1.6 Offset 108h: HPTCV[0-2] - Timer *n* Comparator Value Register

General Behavioral Rules:

- Software can access the various bytes in this register using 32-bit or 64-bit accesses.
- 32-bit accesses can be done to offset 1x8h or 1xCh. 64-bit accesses can be done to 1x8h.
- 32-bit accesses must not be done to 1x9h, 1xAh, 1xBh, 1xDh, 1xEh, or 1xFh.
- Reads to this register return the current value of the comparator.
- If the timer is configured to non-periodic mode:
  - — Writes to this register load the value against which the main counter must be compared for this timer.
  - — When the main counter equals the value last written to this register, the corresponding interrupt can be generated (if so enabled).

  - The value in this register does not change based on the interrupt being generated.
- If the timer is configured to periodic mode:
  - When the main counter equals the value last written to this register, the corresponding interrupt can be generated (if so enabled).
  - After the main counter equals the value in this register, the value in this register is increased by the value last written to the register.
- For example, if the value written to the register is 00000123h, then:
  - An interrupt is generated when the main counter reaches 00000123h.
  - The value in this register is then adjusted by the hardware to 00000246h.
  - Another interrupt is generated when the main counter reaches 00000246h.
  - The value in this register is then adjusted by the hardware to 00000369h.
- As each periodic interrupt occurs, the value in this register increments. When the incremented value is greater than the maximum value possible for this register (FFFFFFFFh for a 32-bit timer or FFFFFFFFFFFFFFFFh for a 64-bit timer), the value wraps around through 0. For example, if the current value in a 32-bit timer is FFFF0000h and the last value written to this register is 20000, then after the next interrupt the value changes to 00010000h.
- Default value for each timer is all 1's for the bits that are implemented. For example, a 32-bit timer has a default value of 00000000FFFFFFFFh. A 64-bit timer has a default value of FFFFFFFFFFFFFFFFh.

Table 32-7. Offset 108h: HPTCV[0-2] - Timer n Comparator Value Register

Description: Timer 0: 108 - 10Fh Timer 1: 128 - 12Fh Timer 2: 148 - 14Fh					
View: IA F	**Base Address:** HPTC			**Offset Start:** 108h at 20h **Offset End:** 10Fh at 20h	
Size: 64 bit	**Default:** Xh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
63 :00	TIMn_COMP	**Timer Compare Value:** Reads to this register return the current value of the comparator. Timers 0, 1, or 2 are configured to non-periodic mode: Writes to this register load the value against which the main counter must be compared for this timer. • When the main counter equals the value last written to this register, the corresponding interrupt can be generated (if so enabled). • The value in this register does not change based on the interrupt being generated. Timer 0 is configured to periodic mode: • When the main counter equals the value last written to this register, the corresponding interrupt can be generated (if so enabled). • After the main counter equals the value in this register, the value in this register is increased by the value last written to the register. • For example, if the value written to the register is 00000123h, then: 1. An interrupt is generated when the main counter reaches 00000123h. 2. The value in this register is then adjusted by the hardware to 00000246h. 3. Another interrupt is generated when the main counter reaches 00000246h. 4. The value in this register is then adjusted by the hardware to 00000369h. • As each periodic interrupt occurs, the value in this register increments. When the incremented value is greater than the maximum value possible for this register (FFFFFFFFh for a 32-bit timer or FFFFFFFFFFFFFFFFh for a 64-bit timer), the value wraps around through zero. For example, if the current value in a 32-bit timer is FFFF0000h and the last value written to this register is 20000, then after the next interrupt the value changes to 00010000h. Default value for each timer is all 1s for the bits that are implemented. For example, a 32-bit timer has a default value of 00000000FFFFFFFFh. A 64-bit timer has a default value of FFFFFFFFFFFFFFFFh.		Xh	RW

32.3 Theory Of Operation

32.3.1 Timer Accuracy Rules

- The timers are expected to be accurate over any 1 ms period to within 0.05% of the time specified in the timer resolution fields.
- Within any 100 microsecond period, the timer is permitted to report a time that is up to two ticks too early or too late. Each tick must be less than or equal to 100 ns; this represents an error of less than 0.2%.
- The timer must be monotonic. It must never return the same value on two consecutive reads (unless the counter has rolled over and actually reached the same value).
- The main counter is clocked by the 14.31818 MHz clock, synchronized into the 66.666 MHz domain. This results in a non-uniform duty cycle on the synchronized clock, but does have the correct average period. The accuracy of the main counter is as accurate as the 14.3818 MHz clock.

32.3.2 Interrupt Mapping

The interrupts associated with the various timers have several interrupt mapping options. When reprogramming the HPET (High Precision Event Timer) interrupt routing scheme (LEG_RT_CNF bit in the General Config Register), a spurious interrupt may occur. This is because the other source of the interrupt (8254 timer) may be asserted. Software must mask interrupts prior to clearing the LEG_RT_CNF bit.

Mapping Option 1: Legacy Replacement Option

In this case, the Legacy Rout bit (LEG_RT_CNF) is set. This forces the mapping found in Table .

Table 32-8. Legacy Replacement Routing

Timer	8259 Mapping	APIC Mapping	Comment
0	IRQ0	IRQ2	In this case, the 8254 timer does not cause any interrupts
1	IRQ8	IRQ8	In this case, the RTC does not cause any interrupts.
2	As per IRQ Routing Field	As per IRQ Routing Field	

Mapping Option 2: Standard Option

In this case, the Legacy Rout bit (LEG_RT_CNF) is zero. Each timer has its own routing control. The interrupts can be routed to various interrupts in the I/O APIC. A capabilities field indicates which interrupts are valid options for the routing.

If a timer is set for edge-triggered mode, the timers must not be shared with any PCI interrupts.

Supported interrupt values are IRQ 20, 21, 22, and 23.

32.3.3 Periodic vs. Non-Periodic Modes

32.3.3.1 Non-Periodic Mode

This mode can be thought of as creating a one-shot.

Timer 0 is configurable to 32-bit or 64-bit mode (default), whereas Timers 1 and 2 only support 32-bit mode.

All three timers support non-periodic mode.

When a timer is set up for non-periodic mode, it generates a value in the main counter that matches the value in the timer's comparator register. Also, another interrupt will be generated when the main counter matches the value in the timer's comparator register after a wrap around.

During run-time, the value in the timer's comparator value register is not changed by the hardware. Software can, of course, change the value.

The Timer 0 Comparator Value register cannot be programmed reliably by a single 64-bit write in a 32-bit environment **except** if only the periodic rate is being changed during run-time. If the actual Timer 0 Comparator Value needs to be reinitialized, then the following software solution always works regardless of the environment:

1. Set the lower 32 bits of the Timer0 Comparator Value register.
2. Set the upper 32 bits of the Timer0 Comparator Value register.

Warning: Software must be careful when programming the comparator registers. If the value written to the register is not sufficiently far in the future, then the counter may pass the value before it reaches the register and the interrupt will be missed. The BIOS will pass a data structure to the OS to indicate that the OS must not attempt to program the periodic timer to a rate faster then X. For the CMI, X is 5 microseconds.

Every timer is required to support the non-periodic mode of operation.

32.3.3.2 Periodic Mode

When a timer is set up for periodic mode, the software writes a value in the timer's comparator value register. When the main counter value matches the value in the timer's comparator value register, an interrupt is generated. The hardware then automatically increases the value in the comparator value register by the last value written to that register.

To make the periodic mode work properly, the main counter is typically written with a value of 0 so that the first interrupt occurs at the right point for the comparator. If the main counter is not set to 0, interrupts may not occur as expected.

During run-time, the value in the timer's comparator value register can be read by software to find out when the next periodic interrupt will be generated (not the rate at which it generates interrupts). Software is expected to remember the last value written to the comparator's value register (the rate at which interrupts are generated).

If software wants to change the periodic rate, it must write a new value to the comparator value register. At the point when the timer's comparator indicates a match, this new value is added to derive the next matching point.

Warning: If the software resets the main counter, the value in the comparator's value register needs to reset as well. This can be done by setting the **TIMERn_VAL_SET_CNF** bit. Again, to avoid race conditions, this must be done with the main counter halted. As the timer period approaches zero, the interrupts associated with the periodic timer may not

get completely serviced before the next timer match occurs. Interrupts may get lost and/or system performance may be degraded in this case.

Each timer is NOT required to support the periodic mode of operation. A capabilities bit indicates if the particular timer supports periodic mode. The reason for this is that supporting the periodic mode adds a significant number of gates.

For CMI, only Timer 0 supports periodic mode. The following usage model is expected:

1. Software clears the ENABLE_CNF bit to prevent any interrupts.
2. Software clears the main counter by writing a value of 00h to it.
3. Software sets the TIMER0_VAL_SET_CNF bit.
4. Software writes the new value in the TIMER0_COMPARATOR_VAL register.
5. Software sets the ENABLE_CNF bit to enable interrupts.

The Timer 0 Comparator Value register cannot be programmed reliably by a single 64-bit write in a 32-bit environment except if only the periodic rate is being changed during run-time. If the actual Timer 0 Comparator Value needs to be reinitialized, then the following software solution will always work regardless of the environment:

1. Set TIMER0_VAL_SET_CNF bit
2. Set the lower 32 bits of the Timer0 Comparator Value register
3. Set TIMER0_VAL_SET_CNF bit
4. Set the upper 32 bits of the Timer0 Comparator Value register

32.3.4 Enabling the Timers

The BIOS or operating system PnP code must rout the interrupts. This includes the Legacy Rout bit, Interrupt Rout bit (for each timer), interrupt type (to select the edge or level type for each timer).

The Device Driver code must do the following for an available timer:

1. Set the Overall Enable bit (Offset 04h, bit 0).
2. Set the timer type field (selects one-shot or periodic).
3. Set the interrupt enable.
4. Set the comparator value.

32.3.5 Interrupt Levels

Interrupts directed to the 8259s are active high. See Chapter 30.0, “Interrupts,” for information regarding the polarity programming of the I/O APIC for detecting internal interrupts.

If the interrupts are mapped to the I/O APIC and set for level-triggered mode, they can be shared with PCI interrupts. If more than one timer is configured to share the same IRQ (using the TIMERn_INT_ROUT_CNF fields), then the software must configure the timers to level-triggered mode. Edge-triggered interrupts cannot be shared.

32.3.6 Handling Interrupts

If each timer has a unique interrupt and the timer has been configured for edge-triggered mode, then there are no specific steps required. No read is required to process the interrupt.

If a timer has been configured to level-triggered mode, then its interrupt must be cleared by the software. This is done by reading the interrupt status register and writing a one back to the bit position for the interrupt to be cleared.

Independent of the mode, software can read the value in the main counter to see how time has passed between when the interrupt was generated and when it was first serviced.

If a timer 0 is set up to generate a periodic interrupt, the software can check to see how much time remains until the next interrupt by checking the timer value register.

32.3.7 Unloading Device Driver Issues

When unloading device drivers for the HPET High Precision Event Timer, some precautions may be needed. For example, if the legacy routing is used, when the HPET High Precision Event Timer is disabled, a spurious interrupt could occur. The OS must mask interrupts prior to clearing the LEG_RT_CNF bit.

§ §

33.0 Serial I/O Unit and Watchdog Timer

33.1 Overview

The Serial I/O unit and Watchdog Timer (SIW) is similar to currently available Super I/O controllers. It is specifically designed for integration into the IICH. It is connected via the LPC bus and currently consists of two UARTs, a Serial Interrupt Controller, a Watchdog Timer and the LPC interface.

33.2 Features

LPC Interface

- Multiplexed command, address and data bus.
- 8-bit I/O transfers.
- 16-bit address qualification for I/O transactions.
- Serial IRQ interface compatible with serialized IRQ support for PCI systems.

Serial Port

- Two Full Function 16550 Compatible Serial Ports.
- Configurable I/O addresses and interrupts.
- 16-Byte FIFOs.
- Supports up to 115 Kbps.
- Programmable Baud Rate Generator.
- Modem Control Circuitry.
- 14.7456 MHz and 48 MHz supported for UART baud clock input.

Watchdog Timer (WDT)

Selectable Prescaler – approximately 1 MHz (1 µs to 1 s) and approximately 1 KHz (1 ms to 10 min).

- 33 MHz Clock (30 ns Clock Ticks).
- Multiple Modes (WDT and Free-Running).
- Free-Running Mode:
  - 1 Stage Timer - Toggles WDT_TOUT# after programmable time.
- WDT Mode:
  - 2 Stage Timer (First stage generates interrupt, second stage drives WDT_TOUT# low).
    First stage generates an SERIRQ, SMI or NMI interrupt (depending on which is enabled) after Programmable time.
    Second stage drives WDT_OUT# low or inverts the previous value.
    Used only after first timeout occurs.

Status bit preserved in RTC well for possible error detection and correction. Drives WDT_TOUT# if OUTPUT is enabled.

- Timer can be disabled (default state) or Locked (Hard Reset required to disable WDT).
- WDT Automatic Reload of Preload value when WDT Reload Sequence is performed.

33.3 Functional Description

33.3.1 Host Processor Interface (LPC)

The host processor communicates with the SIW via the LPC bus. Access is through a series of read/ write registers and accomplished through I/O cycles. All registers are eight bits wide. The SIW registers include global configuration space and device specific regions accessed by setting the Logical Device Number in the SIW Configuration Register 07H (SCR7).

Table 33-1. Address Map

Address	Block Name	Logical Device
04Eh or 20Eh (SIU1_DTR# dependent)	Configuration Index	
04Fh or 20Fh (SIU1_DTR# dependent)	Configuration Data	
Base+(0-7)	Serial Port 1	04H
Base+(0-7)	Serial Port 2	05H
Base+(0-18)	Watchdog Timer	06H

See Section 33.8 for configuration register descriptions and information on setting the base address.

33.4 LPC Interface

The LPC interface is used to control all the logical blocks on the SIW. LPC bus signals use PCI 33 MHz electrical signal characteristics. Refer to the *Low Pin Count (LPC) Interface Specification Rev 1.0.*

33.4.1 LPC Cycles

The following cycle types are supported by the LPC protocol.

Table 33-2. Supported LPC Cycle Types

Cycle Type	Transfer Size
I/O Write	1 Byte
I/O Read	1 Byte

The SIW ignores cycles that it does not support.

33.4.1.1 I/O Read and Write Cycles

The SIW is the target for I/O cycles. I/O cycles are initiated by the host for register or FIFO accesses and generally have minimal synchronization times.

Data transfers are assumed to be exactly 1-byte. If the CPU requested a 16-bit or 32-bit transfer, the host must break it up into 8-bit transfers.

See the *LPC Interface Specification* for the sequence of cycles for the I/O Read and Write cycles.

33.4.2 Policy

The following rules govern the reset policy:

SIW_RESET# is tied to the internal PCI bus reset.

When SIW_RESET# goes active (low):

- The host drives the LFRAME# signal high, tri-states the LAD[3:0] signals.
- The SIU ignores LFRAME#, tri-states the LAD[3:0] pins.

Note: LPC bus signals from SIW are internally tied to the primary LPC interface of the IICH device. Host LPC and SIW LPC names are used interchangeably throughout.

33.4.3 LPC Transfers

33.4.3.1 I/O Transfers

These are generally used for register or FIFO accesses, and generally have minimal synchronization times. The minimum number of wait-states between bytes is one. Data transfers are assumed to be exactly one byte. The host is responsible for breaking up larger data transfers into 8-bit cycles.

Table 33-3. I/O Sync Bits Description

Bits	Indication
0000	Synchronization achieved with no error.
0101	Indicates that synchronization not achieved yet, but the part is driving the bus.
0110	Indicates that synchronization not achieved yet, but the part is driving the bus and expects long synchronization
1010	Special Case: peripheral indicating errors.

33.5 Logical Devices 4 and 5: Serial Ports (UART1 and UART2)

This section describes the Universal Asynchronous Receiver/Transmitter (UART) serial port used for the two UART integrated into the SIW. The UART can be controlled via programmed I/O. The basic programming model is the same for both UARTs with the only difference being the Logical Device Number assigned to each.

The serial port consists of a UART which supports all the functions of a standard 16550 UART including hardware flow control interface.

The UART performs serial-to-parallel conversion on data characters received from a peripheral device or a modem and parallel-to-serial conversion on data characters received from the processor. The processor can read the complete status of the UART at

any time during the functional operation. Available status information includes the type and condition of the transfer operations being performed by the UART and any error conditions (parity, overrun, framing, or break interrupt).

The serial port can operate in either FIFO or non-FIFO mode. In FIFO mode, a 16-byte transmit FIFO holds data from the processor to be transmitted on the serial link and a 16-byte Receive FIFO buffers data from the serial link until read by the processor.

Each UART includes a programmable baud rate generator which is capable of dividing the baud clock input by divisors of one to (2^{16} -1) and producing a 16X clock to drive the internal transmitter and receiver logic. Each UART has complete modem control capability and a processor interrupt system. Interrupts can be programmed to the user's requirements, minimizing the computing required to handle the communications link. Each UART can operate in a polled or an interrupt driven environment as configured by software.

The baud rate generator input is a function of the UART_CLK and a configurable predivide of 1, 8, or 26. See also SIW Configuration (address 29h) in Section 33.8.3.1, “Global Control/Configuration Registers [00h - 2Fh]” on page 1208. The output of the baud rate generator is 16 times the baud rate.

Table 33-4. UART Clock Divider Support

Clock Frequency	14.7456 MHz	48.0 MHz
Predivide Value	8	26
Generator Frequency	1.8432 MHz	1.8462 MHz

Table 33-5. Baud Rate Example

Desired Baud Rate	Divisor	% error @ 1.8432	% error @ 1.8462*
300	384		0.16
1200	96		0.16
2400	48		0.16
4800	24		0.16
9600	12		0.16
19200	6		0.16
38400	3		0.16
56000	2	2.8	3
115200	1		0.16

33.5.1 UART Feature List

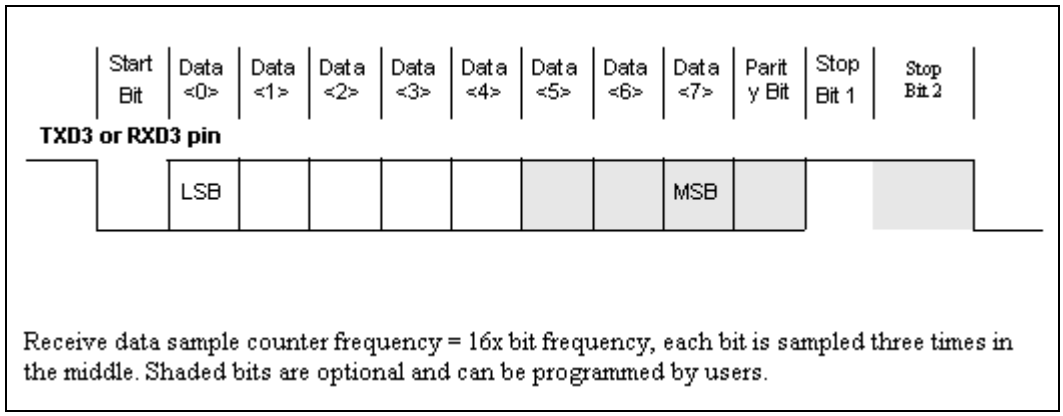
- Functionally compatible with National Semiconductor's* PC16550D.
- Adds or deletes standard asynchronous communications bits (start, stop, and parity) to or from the serial data.
- Independently controlled transmit, receive, line status and data set interrupts.
- Programmable baud rate generator allows division of clock by 1 to (216 -1) and generates an internal 16X clock.
- Modem control functions (CTS#, RTS#, DSR#, DTR#, RI#, and DCD#).

- Fully programmable serial-interface characteristics:
- 5, 6, 7 or 8-bit characters.
- Even, odd, or no parity detection.
- 1, 1-1/2, or 2 stop bit generation.
- Baud rate generation (up to 115 kbps).
- False start bit detection.
- 16-byte Receive FIFO.
- Complete status reporting capability.
- Line break generation and detection.
- Internal diagnostic capabilities include:
  - — Loopback controls for communications link fault isolation.
  - — Break, parity, overrun, and framing error simulation.
  - — Fully prioritized interrupt system controls.

33.5.2 UART Operational Description

The format of a UART data frame is shown in Figure 33-1.

Figure 33-1. Example UART Data Frame



Each data frame is between seven bits and 12 bits long depending on the size of data programmed, if parity is enabled and if two stop bits is selected. The frame begins with a start bit that is represented by a high to low transition. Next, 5 to 8 bits of data are transmitted, beginning with the least significant bit. An optional parity bit follows, which is set if even parity is enabled and an odd number of ones exist within the data byte, or if odd parity is enabled and the data byte contains an even number of ones. The data frame ends with one, one and a half or two stop bits as programmed by the user, which is represented by one or two successive bit periods of a logic one.

The unit is disabled upon reset, the user needs to enable the unit by setting bit six of Interrupt Enable Register. When the unit is enabled, the receiver starts looking for the start bit of a frame; the transmitter starts transmitting data to the transmit data pin if there is data available in the transmit FIFO. Transmit data can be written to the FIFO

before the unit is enabled. When the unit is disabled, the transmitter/receiver finishes the current byte being transmitted/received if it is in the middle of transmitting/ receiving a byte and stops transmitting/receiving more data.

An SIU_RESET# to the SIU forces the internal register and output signals on the serial port to the values listed below.

Table 33-6. UART Register/Signal Reset States

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	RESET	All bits are low.
Interrupt ID Register	RESET	Bit 0 is forced high. Bits 1-3 and 6-7 are forced low. Bits 4-5 are permanently low.
Line Control Register	RESET	All bits are forced low.
Line Status Register	RESET	Bits 0-4,7 are forced low. Bits 5 and 6 are forced high.
Modem Control Register	RESET	Bits 0,1,2,3,4 are forced low. Bits 5,6,7 are permanently low.
Modem Status Register	RESET/Modem signal, read MSR for bits 3-0.	Low
Infrared Selection Register	RESET	All bits are permanently low.
Txd	RESET	High
Int	RESET/ clear LINE STATUS REG	Low
rts_n	RESET	High
dtr_n	RESET	High

33.5.2.1 Programmable Baud Rate Generator

The UART contains a programmable Baud Rate Generator that is capable of taking the UART_CLK input and dividing it by any divisor from 1 to (2^{16} -1). The output frequency of the Baud Rate Generator is 16 times the baud rate. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization to ensure proper operation of the Baud Rate Generator. If both Divisor Latches are loaded with 0, the 16X output clock is stopped. Upon loading either of the Divisor latches, a 16-bit baud counter is immediately loaded. This prevents long counts on initial load. Access to the Divisor latch can be done with a word write.

Note: The UART_CLK is the SIW_CLK input divided by the prescalar set by the SIW Configuration Register (Offset 29h).

The baud rate of the data shifted in/out of the UART is given by:

Baud Rate = UART_CLK(MHz)/[16X Divisor]

For example, if UART_CLK is 14.7456MHz and the divisor is 96, the baud rate is 9600.

A Divisor value of 0 in the Divisor Latch Register is not allowed. The reset value of the divisor is 02.

33.5.3 UART Register Details

There are 12 registers in the UART. These registers share eight address locations in the I/O address space. Table 33-10 shows the registers and their addresses as offsets of a base address. The state of the Divisor Latch Bit (DLAB), which is the MOST significant bit of the Serial Line Control Register, affects the selection of certain of the UART registers. The DLAB bit must be set high by the system software to access the Baud Rate Generator Divisor Latches.

Table 33-7. Summary of UART Registers in I/O Space (DLAB=0)

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: RBR - Receive Buffer Register" on page 1176	00h
00h	00h	"Offset 00h: THR - Transmit Holding Register" on page 1177	00h
01h	01h	"Offset 01h: IER - Interrupt Enable Register" on page 1177	00h

Table 33-8. Summary of UART Registers in I/O Space (DLAB=1)

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: DLL - Programmable Baud Rate Generator Divisor Latch Register Low" on page 1190	02h
01h	01h	"Offset 01h: DLH - Programmable Baud Rate Generator Divisor Latch Register High" on page 1190	00h

Table 33-9. Summary of UART Timer registers in I/O space

Offset Start	Offset End	Register ID - Description	Default Value
02h	02h	"Offset 02h: IIR - Interrupt Identification Register" on page 1179	01h
02h	02h	"Offset 02h: FCR - FIFO Control Register" on page 1180	00h
03h	03h	"Offset 03h: LCR - Line Control Register" on page 1182	00h
04h	04h	"Offset 04h: MCR - Modem Control Register" on page 1184	00h
05h	05h	"Offset 05h: LSR - Line Status Register" on page 1186	60h
06h	06h	"Offset 06h: MSR - Modem Status Register" on page 1189	00h
07h	07h	"Offset 07h: SCR - Scratchpad Register" on page 1190	00h

Table 33-10. Internal Register Descriptions

UART Register Addresses (Base + offset)	DLAB Bit Value	Register Accessed
Base	0	Receive BUFFER (Read-Only)
Base	0	Transmit BUFFER (Write-Only)
Base + 01H	0	Interrupt Enable (Read/Write)
Base + 02H	X	Interrupt I.D. (Read-Only)
Base + 02H	X	FIFO Control (Write-Only)
Base + 03H	X	Line Control (Read/Write)
Base + 04H	X	Modem Control (Read/Write)
Base + 05H	X	Line Status (Read-Only)
Base + 06H	X	Modem Status (Read-Only)
Base + 07H	X	Scratch Pad (Read/Write)
Base	1	Divisor Latch (Lower Byte, Read/Write)
Base + 01H	1	Divisor Latch (Upper Byte, Read/Write)

Note: Base Address for the UART registers listed in Table 33-10 is configurable. See Section 33.8.3, “SIW Configuration Register Summary” on page 1207 for details.

33.5.3.1 Offset 00h: RBR - Receive Buffer Register

In non-FIFO mode, this register holds the character received by the UART's Receive Shift Register. If fewer than eight bits are received, the bits are right-justified and the leading bits are zeroed. Reading the register empties the register and resets the Data Ready (DR) bit in the Line Status Register to zero. Other (error) bits in the Line Status Register are not cleared. In FIFO mode, this register latches the value of the data byte at the top of the FIFO.

Table 33-11. Offset 00h: RBR - Receive Buffer Register

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 0)			**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	RB_7_0	Data byte received (bits [7:0]), least significant bit first		00h	RO

33.5.3.2 Offset 00h: THR - Transmit Holding Register

This register holds the next data byte to be transmitted. When the Transmit Shift Register becomes empty, the contents of the Transmit Holding Register are loaded into the shift register and the transmit data request (TDRQ) bit in the Line Status Register is set to one.

Table 33-12. Offset 00h: THR - Transmit Holding Register

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 0)			**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	TB_7_0	Data byte received (bits [7:0]), least significant bit first		00h	WO

In FIFO mode, writing to THR puts data to the top of the FIFO. The data at the bottom of the FIFO is loaded to the shift register when it is empty.

33.5.3.3 Offset 01h: IER - Interrupt Enable Register

This register enables five types of interrupts which independently activate the int signal and set a value in the Interrupt Identification Register. Each of the five interrupt types can be disabled by resetting the appropriate bit of the IER register. Similarly, by setting the appropriate bits, selected interrupts can be enabled. Receiver time out interrupt can be configured to be separated from the receive data available interrupt (using the bit 5: COMP)

The use of bit 5 to bit 4 is different from the register definition of standard 16550.

Table 33-13. Offset 01h: IER - Interrupt Enable Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 0)			**Offset Start:** 01h **Offset End:** 01h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :06	RSVD	RSVD = 0		00h	RO
05	COMP	**Compatibility Enable:** 0 = Bit 0 of this register also controls RTOIE and bit 4 is RSVD. 1 = Bit 4 of this register controls RTOIE.		0h	RW
04	RTOIE	**Receiver Time Out Interrupt Enable:** 0 = Receiver data Time out interrupt disabled 1 = Receiver data Time out interrupt enabled		0h	RW
03	MIE	**Modem Interrupt Enable:** 0 = Modem Status interrupt disabled 1 = Modem Status interrupt enabled		0h	RW

Table 33-13. Offset 01h: IER - Interrupt Enable Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 0)			**Offset Start:** 01h **Offset End:** 01h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	RLSE	**Receiver Line Status Interrupt Enable:** 0 = Receiver Line Status interrupt disabled 1 = Receiver Line Status interrupt enabled		0h	RW
01	TIE	**Transmit Data request Interrupt Enable:** 0 = Transmit FIFO Data Request interrupt disabled 1 = Transmit FIFO Data Request interrupt enabled		0h	RW
00	RAVIE	**Receiver Data Available Interrupt Enable**: When BIT 5 = 1: 0 = Receiver Data Available (Trigger level reached) interrupt disabled 1 = Receiver Data Available (Trigger level reached) interrupt enabled When BIT 5 = 0: 0 = Receiver data Time Out Interrupt also disabled 1 = Receiver data Time Out Interrupt enabled		0h	RW

33.5.3.4 Offset 02h: IIR - Interrupt Identification Register

In order to minimize software overhead during data character transfers, the UART prioritizes interrupts into four levels (listed in Table 33-14) and records these in the Interrupt Identification Register. The Interrupt Identification Register (IIR) stores information indicating that a prioritized interrupt is pending and the source of that interrupt.

Table 33-14. Interrupt Conditions

Priority Level	Interrupt Origin
1 (highest)	Receiver Line Status. One or more error bits were set.
2	Received Data is available. In FIFO mode, trigger level was reached; in non-FIFO mode, RBR has data.
2	Receiver Time out occurred. It happens in FIFO mode only, when there is data in the receive FIFO but no activity for a time period.
3	Transmitter requests data. In FIFO mode, the transmit FIFO is half or more than half empty; in non-FIFO mode, THR is read already.
4	Modem Status: one or more of the modem input signals has changed state

Table 33-15. Offset 02h: IIR - Interrupt Identification Register

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :06	FIFOES_1_0	**FIFO Mode Enable Status (bits [1:0]):** 00 Non-FIFO mode is selected. 01 Reserved 10 Reserved 11 FIFO mode is selected (TRFIFOE = 1).		00b	RO
05 :04	Reserved	Reserved		00b	
03	TOD_IID3	**Time Out Detected:** 0 = No time out interrupt is pending. 1 = Time out interrupt is pending. (FIFO mode only)		0b	RO
02 :01	IID_2_1	**Interrupt Source Encoded (bits[2:1]):** 00 Modem Status (CTS, DSR, RI, DCD modem signals changed state) 01 Transmit FIFO requests data 10 Received Data Available 11 Receive error (Overrun, parity, framing, break, FIFO error)		00b	RO
00	P_N	**Interrupt Pending:** 0 = Interrupt is pending. (Active low) 1 = No interrupt is pending.		1b	RO

Table 33-16. Interrupt Identification Register Decode (Sheet 1 of 2)

Interrupt ID bits				**Interrupt SET/RESET Function**			
3	2	1	0	**Priority**	**Type**	**Source**	**RESET Control**
0	0	0	1	-	None	No Interrupt is pending.	-
0	1	1	0	Highest	Receiver Line Status	Overrun Error, Parity Error, Framing Error, Break Interrupt.	Reading the Line Status Register.
0	1	0	0	Second Highest	Received Data Available.	Non-FIFO mode: Receive Buffer is full.	Non-FIFO mode: Reading the Receiver Buffer Register.
						FIFO mode: Trigger level was reached.	FIFO mode: Reading bytes until Receiver FIFO drops below trigger level or setting RESETRF bit in FCR register.
1	1	0	0	Second Highest	Character Timeout indication.	FIFO Mode only: At least 1 character is in receiver FIFO and there was no activity for a time period.	Reading the Receiver FIFO or setting RESETRF bit in FCR register.

Table 33-16. Interrupt Identification Register Decode (Sheet 2 of 2)

0	0	1	0	Third Highest	Transmit FIFO Data Request	Non-FIFO mode: Transmit Holding Register Empty	Reading the IIR Register (if the source of the interrupt) or writing into the Transmit Holding Register.
						FIFO mode: Transmit FIFO has half or less than half data.	Reading the IIR Register (if the source of the interrupt) or writing to the Transmitter FIFO.
0	0	0	0	Fourth Highest	Modem Status	Clear to Send, Data Set Ready, Ring Indicator, Received Line Signal Detect	Reading the modem status register

33.5.3.5 Offset 02h: FCR - FIFO Control Register

FCR is a write-only register that is located at the same address as the IIR (IIR is a read-only register). FCR enables/disables the transmitter/receiver FIFOs, clears the transmitter/receiver FIFOs, and sets the receiver FIFO trigger level.

Table 33-17. Offset 02h: FCR - FIFO Control Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :06	ITL_1_0	**Interrupt Trigger Level (bits [1:0]):** When the number of bytes in the receiver FIFO equals the interrupt trigger level programmed into this field and the Received Data Available Interrupt is enabled (via IER), an interrupt is generated and appropriate bits are set in the IIR. 00 1 byte or more in FIFO causes interrupt 01 RSVD 10 8 bytes or more in FIFO causes interrupt 11 RSVD		00b	WO
05 :03	Reserved	Reserved. Must be programmed to 0.		000b	
02	RESETTF	**Reset transmitter FIFO:** When RESETTF is set to 1, the transmitter FIFO counter logic is set to 0, effectively clearing all the bytes in the FIFO. The TDRQ bit in LSR are set and IIR shows a transmitter requests data interrupt if the TIE bit in the IER register is set. The transmitter shift register is not cleared; it completes the current transmission. After the FIFO is cleared, RESETTF is automatically reset to 0. 0 = Writing 0 has no effect 1 = The transmitter FIFO is cleared (FIFO counter set to 0). After clearing, bit is automatically reset to 0		0b	WO

Table 33-17. Offset 02h: FCR - FIFO Control Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	RESETRF	**Reset Receiver FIFO:** When RESETRF is set to 1, the receiver FIFO counter is reset to 0, effectively clearing all the bytes in the FIFO. The DR bit in LSR is reset to 0. All the error bits in the FIFO and the FIFOE bit in LSR are cleared. Any error bits, OE, PE, FE or BI, that had been set in LSR are still set. The receiver shift register is not cleared. If IIR had been set to Received Data Available, it is cleared. After the FIFO is cleared, RESETRF is automatically reset to 0. 0 = Writing 0 has no effect 1 = The receiver FIFO is cleared (FIFO counter set to 0). After clearing, bit is automatically reset to 0		0b	WO
00	TRFIFOE	**Transmit and Receive FIFO Enable:** TRFIFOE enables/disables the transmitter and receiver FIFOs. When TRFIFOE = 1, both FIFOs are enabled (FIFO Mode). When TRFIFOE = 0, the FIFOs are both disabled (non-FIFO Mode). Writing a 0 to this bit clears all bytes in both FIFOs. When changing from FIFO mode to non-FIFO mode and vice versa, data is automatically cleared from the FIFOs. This bit must be 1 when other bits in this register are written or the other bits are not programmed. 0 = FIFOs are disabled 1 = FIFOs are enabled		0b	WO

33.5.3.6 Offset 03h: LCR - Line Control Register

In the Line Control Register (LCR), the system programmer specifies the format of the asynchronous data communications exchange. The serial data format consists of a start bit (logic 0), five to eight data bits, an optional parity bit, and one or two stop bits (logic 1). The LCR has bits for accessing the Divisor Latch and causing a break condition. The programmer can also read the contents of the Line Control Register. The read capability simplifies system programming and eliminates the need for separate storage in system memory.

Table 33-18. Offset 03h: LCR - Line Control Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 03h **Offset End:** 03h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	DLAB	**Divisor register access bit:** This bit is the Divisor Latch Access Bit. It must be set high (logic 1) to access the Divisor Latches of the Baud Rate Generator during a READ or WRITE operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmit Holding Register, or the Interrupt Enable Register. 0 = Access Transmit Holding register (THR), Receive Buffer Register (RBR) and Interrupt Enable Register. 1 = Access Divisor Latch Registers (DLL and DLH).		0b	RW
06	SB	**Set break:** This bit is the set break control bit. It causes a break condition to be transmitted to the receiving UART. When SB is set to a logic 1, the serial output (TXD) is forced to the spacing (logic 0) state and remains there until SB is set to a logic 0. This bit acts only on the TXD pin and has no effect on the transmitter logic. This feature enables the processor to alert a terminal in a computer communications system. If the following sequence is executed, no erroneous characters are transmitted because of the break: Load 00H in the Transmit Holding register in response to a TDRQ interrupt After TDRQ goes high (indicating that 00H is being shifted out), set the break bit before the parity or stop bits reach the TXD pin Wait for the transmitter to be idle (TEMT = 1) and clear the break bit when normal transmission has to be restored During the break, the transmitter can be used as a character timer to accurately establish the break duration. In FIFO mode, wait for the transmitter to be idle (TEMT=1) to set and clear the break bit. 0 = No effect on TXD output 1 = Forces TXD output to 0 (space)		0b	RW

Table 33-18. Offset 03h: LCR - Line Control Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 03h **Offset End:** 03h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	STKYP	**Sticky Parity:** This bit is the "sticky parity" bit, which can be used in multiprocessor communications. When PEN and STKYP are logic 1, the bit that is transmitted in the parity bit location (the bit just before the stop bit) is the complement of the EPS bit. If EPS is 0, then the bit at the parity bit location are transmitted as a 1. In the receiver, if STKYP and PEN are 1, then the receiver compares the bit that is received in the parity bit location with the complement of the EPS bit. If the values being compared are not equal, the receiver sets the Parity Error bit in LSR and causes an error interrupt if line status interrupts were enabled. For example, if EPS is 0, the receiver expects the bit received at the parity bit location to be 1. If it is not, then the parity error bit is set. By forcing the bit value at the parity bit location, rather than calculating a parity value, a system with a master transmitter and multiple receivers can identify some transmitted characters as receiver addresses and the rest of the characters as data. If PEN = 0, STKYP is ignored. 0 = No effect on parity bit 1 = Forces parity bit to be opposite of EPS bit value		0b	RW
04	EPS	**Even parity Select:** This bit is the even parity select bit. When PEN is a logic 1 and EPS is a logic 0, an odd number of logic ones is transmitted or checked in the data word bits and the parity bit. When PEN is a logic 1 and EPS is a logic 1, an even number of logic ones is transmitted or checked in the data word bits and parity bit. If PEN = 0, EPS is ignored. 0 = Sends or checks for odd parity 1 = Sends or checks for even parity		0b	RW
03	PEN	**Parity enable:** This is the parity enable bit. When PEN is a logic 1, a parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The parity bit is used to produce an even or odd number of ones when the data word bits and the parity bit are summed.) 0 = No parity function 1 = Allows parity generation and checking		0b	RW
02	STB	**Stop bits:** This bit specifies the number of stop bits transmitted and received in each serial character. If STB is a logic 0, one stop bit is generated in the transmitted data. If STB is a logic 1 when a 5-bit word length is selected via bits 0 and 1, then 1 and one half stop bits are generated. If STB is a logic 1 when either a 6, 7, or 8-bit word is selected, then two stop bits are generated. The receiver checks the first stop bit only, regardless of the number of stop bits selected. 0 = 1 stop bit 1 = 2 stop bits, except for 5-bit character then 1-1/2 bits		0b	RW
01 :00	WLS_1_0	**Word Length select:** The Word Length Select bits specify the number of data bits in each transmitted or received serial character. 00 5-bit character (default) 01 6-bit character 10 7-bit character 11 8-bit character		00b	RW

33.5.3.7 Offset 04h: MCR - Modem Control Register

This 8-bit register controls the interface with the modem or data set (or a peripheral device emulating a modem).

Table 33-19. Offset 04h: MCR - Modem Control Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 04h **Offset End:** 04h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :05	Reserved	Reserved		000b	
04	LOOP	**Loop back test mode:** This bit provides a local Loopback feature for diagnostic testing of the UART. When LOOP is set to a logic 1, the following occurs: The transmitter serial output is set to a logic 1 state. The OUT2# signal is forced to a logic 1 state. The receiver serial input is disconnected from the pin. The output of the Transmitter Shift register is “looped back” into the receiver shift register input. The four modem control inputs (CTS#, DSR#, DCD#, and RI#) are disconnected from the pins and the modem control output pins (RTS# and DTR#) are forced to their inactive state. • Coming out of the loopback test mode may result in unpredictable activation of the delta bits (bits 3:0) in the Modem Status Register (MSR). It is recommended that MSR be read once to clear the delta bits in the MSR. The lower four bits of the Modem Control register are connected to the upper four Modem Status register bits: • DTR = 1 forces DSR to a 1 • RTS = 1 forces CTS to a 1 • OUT1 = 1 forces RI to a 1 • OUT2= 1 forces DCD to a 1 In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the UART. The transmit, receive and modem control interrupts are operational, except the modem control interrupts are activated by Control register bits, not the modem control inputs. A break signal can also be transferred from the transmitter section to the receiver section in loopback mode. 0 = Normal UART operation 1 = Test mode UART operation		0b	RW
03	OUT2	**OUT2# signal control:** This bit controls the OUT2# output. When the OUT2 bit is set, OUT2# is asserted low. When the OUT2 bit is cleared, OUT2# is deasserted (set high). Outside of the UART module, the OUT2# signal is used to connect the UART’s interrupt output to the Interrupt Controller unit. 0 = OUT2# signal is 1, which disables the UART interrupt. 1 = OUT2# signal is 0.		0b	RW

Table 33-19. Offset 04h: MCR - Modem Control Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 04h **Offset End:** 04h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	OUT1	**Test bit:** This bit is used only in Loopback test mode. See (LOOP) Above.		0b	RW
01	RTS	**Request to Send:** This bit controls the Request to Send (RTS#) output pin. Bit 1 affects the RTS# output in a manner identical to that described below for the DTR bit. 0 = RTS# pin is 1 1 = RTS# pin is 0		0b	RW
00	DTR	**Data Terminal Ready:** This bit controls the Data Terminal Ready output. When bit 0 is set to a logic 1, the DTR# output is force to a logic 0. When bit 0 is reset to a logic 0, the DTR# output pin is forced to a logic 1. • The DTR# output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding modem or data set. 0 = DTR# pin is 1 1 = DTR# pin is 0		0b	RW

33.5.3.8 Offset 05h: LSR - Line Status Register

This register provides status information to the processor concerning the data transfers. Bits 5 and 6 show information about the transmitter section. The rest of the bits contain information about the receiver.

In non-FIFO mode, three of the LSR register bits, parity error, framing error, and break interrupt, show the error status of the character that has just been received. In FIFO mode, these three bits of status are stored with each received character in the FIFO. LSR shows the status bits of the character at the top of the FIFO. When the character at the top of the FIFO has errors, the LSR error bits are set and are not cleared until software reads LSR, even if the character in the FIFO is read and a new character is now at the top of the FIFO.

Bits one through four are the error conditions that produce a receiver line status interrupt when any of the corresponding conditions are detected and the interrupt is enabled. These bits are not cleared by reading the erroneous byte from the FIFO or receive buffer. They are cleared only by reading LSR. In FIFO mode, the line status interrupt occurs only when the erroneous byte is at the top of the FIFO. If the erroneous byte being received is not at the top of the FIFO, an interrupt is generated only after the previous bytes are read and the erroneous byte is moved to the top of the FIFO.

Table 33-20. Offset 05h: LSR - Line Status Register (Sheet 1 of 3)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 05h **Offset End:** 05h	
Size: 8 bit	**Default:** 60h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	FIFOE	**FIFO Error Status:** This bit is reset only when all the error bytes have been read from the FIFO. A processor read to the Line Status register does not reset this bit. Non-FIFO mode: 0 = Bit is always "0" indicating no FIFO. FIFO mode: 0 = All error bytes have been read from the FIFO 1 = At least one character in the receiver FIFO contains a parity error, framing error, or break indication. When DMA requests are enabled (IER bit7 is set to 1) and FIFOE is set to 1, no receive DMA request is generated even though the receive FIFO reaches the trigger level and the error interrupt is generated. When DMA requests are not enabled (IER bit7 is set to 0), FIFOE set to 1 does not generate interrupt.		0b	RO
06	TEMT	**Transmitter Empty:** Non-FIFO mode: 0 = Either the Transmit Holding register or the Tansmitter Shift register contain a data character. 1 = The Transmit Holding register and the Transmitter Shift register are both empty. FIFO mode: 0 = Either the transmitter FIFO or the Transmit Shift register contain a data character. 1 = The transmitter FIFO and the Transmit Shift register are both empty.		1b	RO
05	TRDQ	**Transmit Data Request:** TDRQ indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the processor when the transmit data request interrupt enable is set high. Non-FIFO mode: 0 = No character transferred from the Transmit Holding register into the Transmit Shift register. 1 = A character has transferred from the Transmit Holding register into the Transmit Shift register. ***Note:*** Bit is reset to logic 0 with the loading of the Transmit Holding register by the processor. FIFO mode: 0 = When at least one byte is written to the transmit FIFO. When more than 16 characters are loaded into the FIFO, the excess characters are lost. 1 = Transmit FIFO is empty or the RESETTF bit in FCR, has been set to 1.		1b	RO

Table 33-20. Offset 05h: LSR - Line Status Register (Sheet 2 of 3)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 05h **Offset End:** 05h	
Size: 8 bit	**Default:** 60h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	BI	**Break Interrupt:** BI is set to a logic 1 when the received data input is held in the spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + parity bit + stop bits).The BI is reset to a logic "0" when the processor reads the Line Status register. 0 = No Break signal has been received. 1 = Break signal occurred. In FIFO mode, only one character (equal to 00H), is loaded into the FIFO regardless of the length of the break condition. BI shows the break condition for the character at the top of the FIFO, not the most recently received character.		0h	RO
03	FE	**Framing Error:** FE indicates that the received character did not have a valid stop bit. This bit is reset to a logic "0" when the processor reads the Line Status Register. 0 = No Framing error. 1 = Invalid stop bit has been detected. FE is set to a logic 1 when the bit following the last data bit or parity bit is detected as a logic 0 (spacing level). If the Line Control register had been set for two stop bit mode, the receiver does not check for a valid second stop bit. The FE indicator is reset when the processor reads the Line Status Register. The UART resynchronizes after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this "start" bit twice and then takes in the "data". In FIFO mode FE shows a Framing error for the character at the top of the FIFO, not for the most recently received character.		0h	RO
02	PE	**Parity Error:** PE indicates that the received data character does not have the correct even or odd parity, as selected by the even parity select bit. The PE is set to logic 1 upon detection of a parity error and is reset to a logic 0 when the processor reads the Line Status register. In FIFO mode, PE shows a parity error for the character at the top of the FIFO, not the most recently received character. 0 = No Parity error. 1 = Parity error has occurred.		0h	RO

Table 33-20. Offset 05h: LSR - Line Status Register (Sheet 3 of 3)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 05h **Offset End:** 05h	
Size: 8 bit	**Default:** 60h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	OE	**Overrun Error:** In non-FIFO mode, OE indicates that data in the receiver buffer register was not read by the processor before the next character was transferred into the receiver buffer register, thereby destroying the previous character. In FIFO mode, OE indicates that all 16 bytes of the FIFO are full and the most recently received byte has been discarded. The OE indicator is set to a logic "1" upon detection of an overrun condition and reset when the processor reads the Line Status Register. 0 = No data has been lost 1 = Received data has been lost.		0h	RO
00	DR	**Data Ready:** DR is set to logic 1 when complete incoming character has been received and transferred into the Receiver Buffer Register (RBR) or the FIFO. In non-FIFO mode, DR is reset to 0 when the receive buffer is read. In FIFO mode, DR is reset to a logic 0 if the FIFO is empty (last character has been read from Receiver Buffer Register) or the RESETRF bit is set in FCR. 0 = No data has been received 1 = Data available in RBR or the FIFO.		0h	RO

33.5.3.9 Offset 06h: MSR - Modem Status Register

This 8-bit register provides the current state of the control lines from the modem or data set (or a peripheral device emulating a modem) to the processor. In addition to this current state information, four bits of the Modem Status register provide change information. Bits 03:00 are set to a logic 1 when a control input from the Modem changes state. They are reset to a logic 0 when the processor reads the Modem Status register.

When bits 0, 1, 2, or 3 are set to logic 1, a Modem Status interrupt is generated if bit 3 of the Interrupt Enable Register is set.

Table 33-21. Offset 06h: MSR - Modem Status Register

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 06h **Offset End:** 06h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	DCD	**Data Carrier Detect:** This bit is the complement of the Data Carrier Detect (DCD#) input. This bit is equivalent to bit OUT2 of the Modem Control register if LOOP in the MCR is set to 1. 0 = DCD# pin is 1 1 = DCD# pin is 0		0b	RO
06	RI	**Ring Indicator:** This bit is the complement of the ring Indicator (RI#) input. This bit is equivalent to bit OUT1 of the Modem Control register if LOOP in the MCR is set to 1. 0 = RI# pin is 1 1 = RI# pin is 0		0b	RO
05	DSR	**Data Set Ready:** This bit is the complement of the Data Set Ready (DSR#) input. This bit is equivalent to bit DTR of the Modem Control register if LOOP in the MCR is set to 1. 0 = DSR# pin is 1 1 = DSR# pin is 0		0b	RO
04	CTS	**Clear to Send:** This bit is the complement of the Clear to Send (CTS#) input. This bit is equivalent to bit RTS of the Modem Control register if LOOP in the MCR is set to 1. 0 = CTS# pin is 1 1 = CTS# pin is 0		0b	RO
03	DDCD	**Delta Data Carrier Detect:** 0 = No change in DCD# pin since last read of MSR 1 = DCD# pin has changed state		0b	RO
02	TERI	**Trailing Edge Ring Indicator:** 0 = RI# pin has not changed from 0 to 1 since last read of MSR 1 = RI# pin has changed from 0 to 1		0b	RO
01	DDSR	**Delta Data Set Ready:** 0 = No change in DSR# pin since last read of MSR 1 = DSR# pin has changed state		0b	RO
00	DCTS	**Delta Clear To Send:** 0 = No change in CTS# pin since last read of MSR 1 = CTS# pin has changed state		0b	RO

33.5.3.10 Offset 07h: SCR - Scratchpad Register

This 8-bit read/write register has no effect on the UART. It is intended as a scratchpad register for use by the programmer.

Table 33-22. Offset 07h: SCR - Scratchpad Register

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 07h **Offset End:** 07h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	SP_7_0	No effect on UART functionality		00h	RW

33.5.3.11 Offset 00h: DLL - Programmable Baud Rate Generator Divisor Latch Register Low

See Section 33.5.2.1, “Programmable Baud Rate Generator” on page 1174.

Table 33-23. Offset 00h: DLL - Programmable Baud Rate Generator Divisor Latch Register Low

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 1)			**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** 02h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	BR_7_0	Low byte compare value to generate baud rate		02h	RW

33.5.3.12 Offset 01h: DLH - Programmable Baud Rate Generator Divisor Latch Register High

See Section 33.5.2.1, “Programmable Baud Rate Generator” on page 1174.

Table 33-24. Offset 01h: DLH - Programmable Baud Rate Generator Divisor Latch Register High

Description:					
View: IA F	**Base Address:** Base (IO) (DLAB = 1)			**Offset Start:** 01h **Offset End:** 01h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 :00	BR_15_8	High byte compare value to generate baud rate		00h	RW

33.5.4 FIFO Operation

33.5.4.1 FIFO Interrupt Mode Operation

33.5.4.1.1 Receiver Interrupt

When the Receive FIFO and receiver interrupts are enabled (FCR[0]=1 and IER[0]=1), receiver interrupts occur as follows:

- The receive data available interrupt is invoked when the FIFO has reached its programmed trigger level. The interrupt is cleared when the FIFO drops below the programmed trigger level.
- The IIR receive data available indication also occurs when the FIFO trigger level is reached, and like the interrupt, the bits are cleared when the FIFO drops below the trigger level.
- The receiver line status interrupt (IIR = C6H), as before, has the highest priority. The receiver data available interrupt (IIR=C4H) is lower. The line status interrupt occurs only when the character at the top of the FIFO has errors.
- The data ready bit (DR in LSR register) is set to 1 as soon as a character is transferred from the shift register to the Receive FIFO. This bit is reset to 0 when the FIFO is empty.

33.5.4.1.2 Character Timeout Interrupt

When the receiver FIFO and receiver time out interrupt are enabled, a character timeout interrupt occurs when all of the following conditions exist:

- At least one character is in the FIFO.
- The last received character was longer than four continuous character times ago (if two stop bits are programmed the second one is included in this time delay).
- The most recent processor read of the FIFO was longer than four continuous character times ago.
- The receive FIFO trigger level is greater than one.

The maximum time between a received character and a timeout interrupt is 160 ms at 300 baud with a 12-bit receive character (i.e., one start, eight data, one parity, and two stop bits).

When a time out interrupt occurs, it is cleared and the timer is reset when the processor reads one character from the receiver FIFO. If a timeout interrupt has not occurred, the timeout timer is reset after a new character is received or after the processor reads the receiver FIFO.

33.5.4.1.3 Transmit Interrupt

When the transmitter FIFO and transmitter interrupt are enabled (FCR[0]=1, IER[1]=1), transmit interrupts occur as follows:

The transmitter holding register interrupt occurs when the transmit FIFO is empty; it is cleared as soon as the transmitter holder register is written to (1 to 16 characters may be written to the transmit FIFO while servicing this interrupt) or the IIR is read.

The transmitter FIFO empty indications are delayed one character time minus the last stop bit time whenever the following occurs: THRE = 1 and there have not been at least two bytes at the same time in the transmit FIFO since the last THRE = 1. The first transmitter interrupt after changing FCRO is immediate if it is enabled.

33.5.4.2 FIFO Polled Mode Operation

With the FIFOs enabled (TRFIFOE bit of FCR set to 1), setting IER[3:0] to all zeros puts the serial port in the FIFO polled mode of operation. Since the receiver and the transmitter are controlled separately, either one or both can be in the polled mode of operation. In this mode, software checks receiver and transmitter status via the LSR. As stated in the register description:

- LSR[0] is set as long as there is one byte in the receiver FIFO.
- LSR[1] through LSR[4] specify which error(s) has occurred for the character at the top of the FIFO. Character error status is handled the same way as interrupt mode. The IIR is not affected since IER[2] = 0.
- LSR[5] indicates when the transmitter FIFO needs data.
- LSR[6] indicates that both the transmitter FIFO and shift register are empty.
- LSR[7] indicates whether there are any errors in the receiver FIFO.

33.6 Logical Device 6: Watchdog Timer

33.6.1 Overview

This device is a Watchdog timer that provides a resolution that ranges from 1 µs to 10 minutes. The timer uses a 35-bit down-counter.

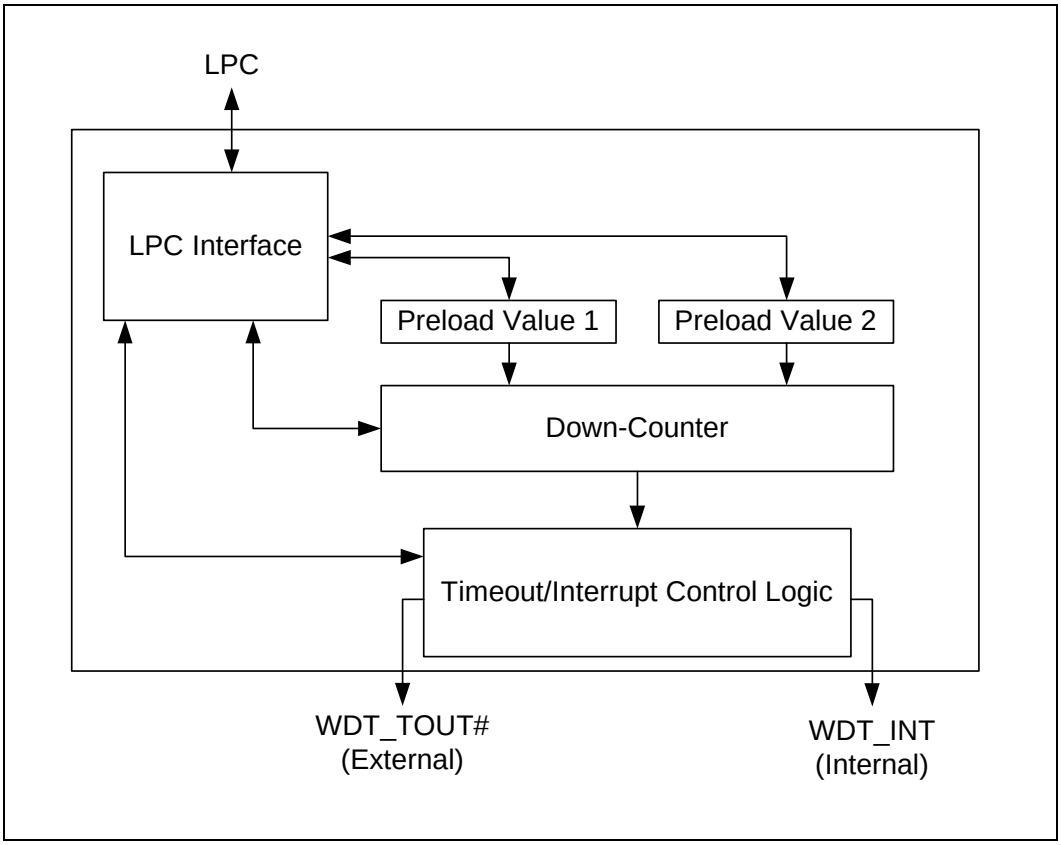
The counter is loaded with the value from the 1st Preload register. The timer is then enabled and it starts counting down. The time at which the WDT first starts counting down is called the first stage. If the host fails to reload the WDT before the 35-bit down counter reaches zero the WDT generates an internal interrupt.

After the interrupt is generated the WDT loads the value from the 2nd Preload register into the WDT's 35-bit Down-Counter and starts counting down. The WDT is now in the second stage. If the host still fails to reload the WDT before the second timeout, the WDT drives the WDT_TOUT# pin low and sets the timeout bit (WDT_TIMEOUT). This bit indicates that the System has become unstable. The WDT_TOUT# pin is held low until the system is Reset or the WDT times out again (Depends on TOUT_CNF). The process of reloading the WDT involves the following sequence of writes:

1. Write “80” to offset BAR1 + 0Ch
2. Write “86” to offset BAR1 + 0Ch
3. Write ‘1’ to WDT_RELOAD in Reload Register.

The same process is used for setting the values in the preload registers. The only difference exists in step 3. Instead of writing a ‘1’ to the WDT_RELOAD, you write the desired preload value into the corresponding Preload register. This value is not loaded into the 35-bit down counter until the next time the WDT reenters the stage. For example, if Preload Value 2 is changed, it is not loaded into the 35-bit down counter until the next time the WDT enters the second stage.

Figure 33-2. WDT Block Diagram



33.6.2 Watchdog Timer Register Details

All registers not mentioned are reserved.

Table 33-25. Summary of Watchdog Timer Registers in I/O Space

Offset Start	Offset End	Register ID - Description	Default Value
00h	00h	"Offset 00h: PV1R0 - Preload Value 1 Register 0" on page 1194	FFh
01h	01h	"Offset 01h: PV1R1 - Preload Value 1 Register 1" on page 1195	FFh
02h	02h	"Offset 02h: PV1R2 - Preload Value 1 Register 2" on page 1195	0Fh
04h	04h	"Offset 04h: PV2R0 - Preload Value 2 Register 0" on page 1196	FFh
05h	05h	"Offset 05h: PV2R1 - Preload Value 2 Register 1" on page 1196	FFh
06h	06h	"Offset 06h: PV2R2 - Preload Value 2 Register 2" on page 1197	0Fh
08h	08h	"Offset 08h: GISR - General Interrupt Status Register" on page 1197	00h
0Ch	0Ch	"Offset 0Ch: RR0 - Reload Register 0" on page 1198	00h
0Dh	0Dh	"Offset 0Dh: RR1 - Reload Register 1" on page 1199	00h
10h	10h	"Offset 10h: WDTCR - WDT Configuration Register" on page 1199	00h
18h	18h	"Offset 18h: WDTLR - WDT Lock Register" on page 1201	00h

Note: Base Address for the Watchdog Timer registers, listed in this section, is configurable. See Section 33.8.3, "SIW Configuration Register Summary" on page 1207 for details.

33.6.2.1 Offset 00h: PV1R0 - Preload Value 1 Register 0

Table 33-26. Offset 00h: PV1R0 - Preload Value 1 Register 0

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 00h **Offset End:** 00h	
Size: 8 bit	**Default:** FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PLOAD1_7_0	**Preload_Value_1 [7:0]:** This register is used to hold the bits 0 through 7 of the preload value 1 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the first stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e. zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		FFh	RW

33.6.2.2 Offset 01h: PV1R1 - Preload Value 1 Register 1

Table 33-27. Offset 01h: PV1R1 - Preload Value 1 Register 1

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 01h **Offset End:** 01h	
Size: 8 bit	**Default:** FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PLOAD1_15_8	**Preload_Value_1 [15:8]:** This register is used to hold the bits 8 through 15 of the preload value 1 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the first stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e. zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		FFh	RW

33.6.2.3 Offset 02h: PV1R2 - Preload Value 1 Register 2

Table 33-28. Offset 02h: PV1R2 - Preload Value 1 Register 2

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 02h **Offset End:** 02h	
Size: 8 bit	**Default:** 0Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03 : 00	PLOAD_19_16	**Preload_Value_1 [19:16]:** This register is used to hold the bits 16 through 19 of the preload value 1 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the first stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e. zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		Fh	RW

33.6.2.4 Offset 04h: PV2R0 - Preload Value 2 Register 0

Table 33-29. Offset 04h: PV2R0 - Preload Value 2 Register 0

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 04h **Offset End:** 04h	
Size: 8 bit	**Default:** FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PLOAD2_7_0	**Preload_Value_2 [7:0]:** This register is used to hold the bits 0 through 7 of the preload value2 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the second stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e., zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		FFh	RW

33.6.2.5 Offset 05h: PV2R1 - Preload Value 2 Register 1

Table 33-30. Offset 05h: PV2R1 - Preload Value 2 Register 1

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 05h **Offset End:** 05h	
Size: 8 bit	**Default:** FFh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PLOAD2_15_8	**Preload_Value_2 [15:8]:** This register is used to hold the bits 8 through 15 of the preload value2 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the second stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e., zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		FFh	RW

33.6.2.6 Offset 06h: PV2R2 - Preload Value 2 Register 2

Table 33-31. Offset 06h: PV2R2 - Preload Value 2 Register 2

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 06h **Offset End:** 06h	
Size: 8 bit	**Default:** 0Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	Reserved	Reserved		0h	
03 : 00	PLOAD2_19_16	**Preload_Value_2 [19:16]:** This register is used to hold the bits 16 through 19 of the preload value2 for the WDT Timer. The Value in the Preload Register is automatically transferred into the 35-bit down counter every time the WDT enters the second stage. The value loaded into the preload register needs to be one less than the intended period. This is because the timer makes use of zero-based counting (i.e. zero is counted as part of the decrement). Refer to Section 33.6.3.2 for details on how to change the value of this register.		Fh	RW

33.6.2.7 Offset 08h: GISR - General Interrupt Status Register

Table 33-32. Offset 08h: GISR - General Interrupt Status Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		00h	
02	SMIACT	**Watchdog Timer SMI Interrupt Active (1st Stage):** This bit is set when the first Stage of the 35-bit Down Counter Reaches zero. An SMI interrupt is generated if WDT_INT_TYPE is configured to do so (See WDT Configuration Register). This is a sticky bit and is only cleared by writing a '1'. This SMI interrupt will be routed to GPIO6 so that BIOS can use the existing SMM handler to service this interrupt. 0 = No Interrupt 1 = Interrupt Active ***Note:*** This bit is not set in free running mode. Also note that to route the SMI interrupt to GPI6, the GPI_ROUT[13:12] register in D31:F0:B8h must be set to "01" to generate the WDT SMI interrupt and GPIO6 cannot be used as a general purpose input pin.		0h	RWC

Table 33-32. Offset 08h: GISR - General Interrupt Status Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	NMIACT	**Watchdog Timer NMI Interrupt Active (1st Stage):** This bit is set when the first Stage of the 35-bit Down Counter Reaches zero. An NMI interrupt is generated if WDT_INT_TYPE is configured to do so (See WDT Configuration Register). This is a sticky bit and is only cleared by writing a '1'. 0 = No Interrupt 1 = Interrupt Active ***Note:*** This bit is not set in free running mode.		0h	RWC
00	SERIRQACT	**Watchdog Timer SERIRQ Interrupt Active (1st Stage):** This bit is set when the first Stage of the 35-bit Down Counter Reaches zero. An SERIRQ interrupt is generated if WDT_INT_TYPE is configured to do so (See WDT Configuration Register). This is a sticky bit and is only cleared by writing a '1'. 0 = No Interrupt 1 = Interrupt Active ***Note:*** This bit is not set in free running mode.		0h	RWC

33.6.2.8 Offset 0Ch: RR0 - Reload Register 0

Table 33-33. Offset 0Ch: RR0 - Reload Register 0

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 0Ch **Offset End:** 0Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	Reserved	**Reserved.** Must be programmed to 0.		00h	

33.6.2.9 Offset 0Dh: RR1 - Reload Register 1

Table 33-34. Offset 0Dh: RR1 - Reload Register 1

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 0Dh **Offset End:** 0Dh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved	Reserved		00h	
01	TOUT	**WDT_TIMEOUT:** This bit is located in the RTC Well and it's value is not lost if the host resets the system. It is set to '1' if the host fails to reset the WDT before the 35-bit Down-Counter reaches zero for the second time in a row. This bit is cleared by performing the Register Unlocking Sequence followed by a '1' to this bit. 0 = Normal (Default) 1 = System has become unstable. ***Note:*** In free running mode this bit is set every time the down counter reaches zero.		0h	RW
00	RELOAD	**WDT_RELOAD:** To prevent a timeout the host must perform the Register Unlocking Sequence followed by a '1' to this bit. Refer to Section 33.6.3.2 for details on how to change the value of this register.		0h	RW

33.6.2.10 Offset 10h: WDTCR - WDT Configuration Register

Table 33-35. Offset 10h: WDTCR - WDT Configuration Register (Sheet 1 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 10h **Offset End:** 10h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 06	Reserved	Reserved		00h	
05	WDT_TOUT_EN	**WDT Timeout Output Enable:** This bit indicates whether or not the WDT toggles the external WDT_TOUT# pin if the WDT times out. 0 = Enabled (Default) 1 = Disabled		0h	RW

Table 33-35. Offset 10h: WDTCR - WDT Configuration Register (Sheet 2 of 2)

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 10h **Offset End:** 10h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04 : 03	Reserved	Reserved		00h	
02	WDT_PRE_SEL	**WDT Prescaler Select:** The WDT provides two options for prescaling the main Down Counter. The preload values are loaded into the main down counter right justified. The prescaler adjusts the starting point of the 35-bit down counter. 0 = The 20-bit Preload Value is loaded into bits 34:15 of the main down counter. The resulting timer clock is the PCI Clock (33 MHz) divided by 2^{15}. The approximate clock generated is 1 KHz, (1 ms to 10 min). (Default) 1 = The 20-bit Preload Value is loaded into bits 24:05 of the main down counter. The resulting timer clock is the PCI Clock (33 MHz) divided by 2^{5}. The approximate clock generated is 1 MHz, (1 µs to 1sec)		0h	RW
01 : 00	WDT_INT_TYPE	**WDT_INT_TYPE:** The WDT timer supports programmable routing of interrupts. The set of bits allows the user to choose the type of interrupt desired if the WDT reached the end of the first stage without being reset. The interrupt status is reported in the WDT General Interrupt Status register. 00 SERIRQ (Default) 01 NMI 10 SMI 11 Disabled ***Note:*** SERIRQ is Active Low		00h	RW

33.6.2.11 Offset 18h: WDTLR - WDT Lock Register

Table 33-36. Offset 18h: WDTLR - WDT Lock Register

Description:					
View: IA F	**Base Address:** Base (IO)			**Offset Start:** 18h **Offset End:** 18h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved	Reserved		0h	
02	WDT_TOUT_CNF	**WDT Timeout Configuration:** This register is used to choose the functionality of the timer. 0 = Watchdog Timer Mode: When enabled (i.e. WDT_ENABLE goes from '0' to '1') the timer reloads Preload Value 1 and start decrementing. (Default) Upon reaching the second stage timeout the WDT_TOUT# is driven low once and does not change again until Power is cycled or a hard reset occurs. 1 = Free Running Mode: WDT_TOUT# changes from previous state when the next timeout occurs. The timer ignores the first stage. The timer only uses Preload Value 2. In this mode the timer is restarted whenever WDT_ENABLE goes from a 0 to a 1. This means that the timer reloads Preload Value 2 and start decrementing every time it is enabled. In free running mode it is not necessary to reload the timer as it is done automatically every time the descrementer reaches zero.		0h	RW
01	WDT_ENABLE	**Watchdog Timer Enable:** The following bit enables or disables the WDT. 0 = Disabled (Default) 1 = Enabled ***Note:*** This bit cannot be modified if WDT_LOCK has been set. ***Note:*** In free-running mode Preload Value 2 is reloaded into the down counter every time WDT_ENABLE goes from '0' to '1'. ***Note:*** In WDT mode Preload Value 1 is reloaded every time WDT_ENABLE goes from '0' to '1' or the WDT_RELOAD bit is written using the proper sequence of writes (See Register Unlocking Sequence). When the WDT second stage timeout occurs, a reset must happen. ***Note:*** Software must guarantee that a timeout is not about to occur before disabling the timer. A reload sequence is suggested.		0h	RW
00	WDT_LOCK	**Watchdog Timer Lock:** Setting this bit locks the values of this register until a hard-reset occurs or power is cycled. 0 = Unlocked (Default) 1 = Locked ***Note:*** Writing a "0" has no effect on this bit. Write is only allowed from "0" to "1" once. It cannot be changed until either power is cycled or a hard-reset occurs.		0h	RWL

33.6.3 Theory Of Operation

33.6.3.1 RTC Well and WDT_TOUT# Functionality

The WDT_TIMEOUT bit is set to a '1' when the WDT 35-bit down counter reaches zero for the second time in a row. Then the WDT_TOUT# pin is toggled LOW by the WDT from the IICH. The board designer must attach the WDT_TOUT# to the appropriate external signal. If WDT_TOUT_CNF is a '1' the WDT toggles WDT_TOUT# again the next time a time out occurs. Otherwise WDT_TOUT# is driven low until the system is reset or power is cycled.

33.6.3.2 Register Unlocking Sequence

The register unlocking sequence is necessary whenever writing to the RELOAD register or either PRELOAD_VALUE registers. The host must write a sequence of two writes to offset BAR1 + 0Ch before attempting to write to either the WDT_RELOAD and WDT_TIMEOUT bits of the RELOAD register or the PRELOAD_VALUE registers. The first writes are "80" and "86" (in that order) to offset BAR1 + 0Ch. The next write is to the proper memory mapped register (e.g., RELOAD, PRELOAD_VALUE_1, PRELOAD_VALUE_2). Any deviation from the sequence (writes to memory-mapped registers) causes the host to have to restart the sequence.

When performing register unlocking, software must issue the cycles using byte access only. Otherwise the unlocking sequence will not work properly.

The following is an example of how to prevent a timeout:

1. Write "80" to offset BAR1 + 0Ch.
2. Write "86" to offset BAR1 + 0Ch.
3. Write a '1' to RELOAD [8] (WDT_RELOAD) of the Reload Register.

Note: Any subsequent writes require that this sequence be performed again.

33.6.3.3 Reload Sequence

To keep the timer from causing an interrupt or driving WDT_TOUT#, the timer must be updated periodically. Other timers refer to "updating the timer" as "kicking the timer". The frequency of updates required is dependent on the value of the Preload values. To update the timer the Register Unlocking Sequence must be performed followed by writing a '1' to bit 8 at offset BAR1 + 0Ch within the watchdog timer memory mapped space. This sequence of events is referred to as the "Reload Sequence".

33.6.3.4 Low Power State

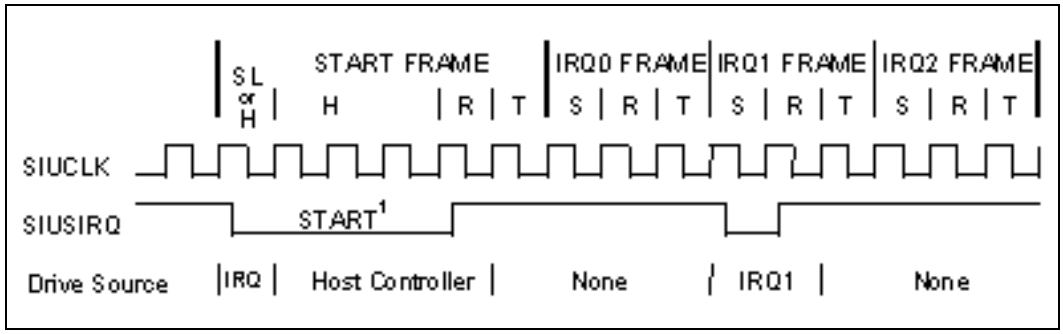
The Watchdog Timer does not operate when PCICLK is stopped.

33.7 Serial IRQ

The SIW supports the serial interrupt to transmit interrupt information to the host system. The serial interrupt scheme adheres to the *Serial IRQ Specification*.

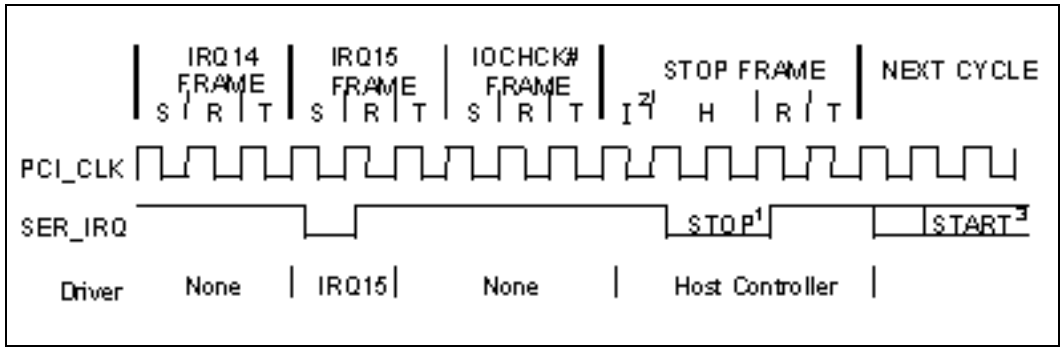
33.7.1 Timing Diagrams For SIW_SERIRQ Cycle

Figure 33-3. Start Frame Timing with Source Sampled a Low Pulse on IRQ1



Notes:
1. H=Host Control; R=Recovery; T=Turn-Around; SL=Slave Control; S=Sample
2. Start Frame pulse can be 4-8 clocks wide depending on the location of the device in the PCI bridge hierarchy in a synchronous bridge design.

Figure 33-4. Stop Frame Timing with Host Using Quiet Mode Sampling Period



Notes:
1. H=Host Control; R=Recovery; T=Turn-Around; S=Sample; I=Idle
2. Stop pulse is two clocks wide for Quiet mode, three clocks wide for Continuous mode.
3. There may be none, one or more Idle states during the Stop Frame.
4. The next SIW_SERIRQ cycle's Start Frame pulse may or may not start immediately after the turn-around clock of the Stop Frame.

33.7.1.1 SIW_SERIRQ Cycle Control

There are two modes of operation for the SIW_SERIRQ Start Frame.

1. **Quiet (Active) Mode**: Any device may initiate a Start Frame by driving the SIW_SERIRQ low for one clock, while the SIW_SERIRQ is Idle. After driving low for one clock the SIW_SERIRQ is immediately tri-stated without at any time driving high. A Start Frame may not be initiated while the SIW_SERIRQ is Active. The SIW_SERIRQ is Idle between Stop and Start Frames. The SIW_SERIRQ is Active

between Start and Stop Frames. This mode of operation allows the SIW_SERIRQ to be Idle when there are no IRQ/Data transitions which should be most of the time.

Once a Start Frame has been initiated the Host Controller takes over driving the SIW_SERIRQ low in the next clock and continues driving the SIW_SERIRQ low for a programmable period of three to seven clocks. This makes a total low pulse width of four to eight clocks. Finally, the Host Controller drives the SIW_SERIRQ back high for one clock, then tri-state.

Any SIW_SERIRQ Device (i.e., The SIU and WDT) which detects any transition on an IRQ/Data line for which it is responsible must initiate a Start Frame in order to update the Host Controller unless the SIW_SERIRQ is already in an SIW_SERIRQ Cycle and the IRQ/Data transition can be delivered in that SIW_SERIRQ Cycle.

2. **Continuous (Idle) Mode**: Only the Host controller can initiate a Start Frame to update IRQ/ Data line information. All other SIW_SERIRQ agents become passive and may not initiate a Start Frame. SIW_SERIRQ is driven low for four to eight clocks by Host Controller. This mode has two functions. It can be used to stop or idle the SIW_SERIRQ or the Host Controller can operate SIW_SERIRQ in a continuous mode by initiating a Start Frame at the end of every Stop Frame.

   An SIW_SERIRQ mode transition can only occur during the Stop Frame. Upon reset, SIW_SERIRQ bus is defaulted to Continuous mode, therefore only the Host controller can initiate the first Start Frame. Slaves must continuously sample the Stop Frames pulse width to determine the next SIW_SERIRQ Cycle's mode.

33.7.1.2 SIW_SERIRQ Data Frame

Once a Start Frame has been initiated, the SIW watches for the rising edge of the Start Pulse and start counting IRQ/Data Frames from there. Each IRQ/Data Frame is three clocks: Sample phase, Recovery phase, and Turn-around phase. During the Sample phase the SIW drives the SIW_SERIRQ low, if and only if, its last detected IRQ/Data value was low. If its detected IRQ/Data value is high, SIW_SERIRQ is left tri-stated. During the Recovery phase the SIW drives the SIW_SERIRQ high, if and only if, it had driven the SIW_SERIRQ low during the previous Sample Phase. During the Turn-around Phase the SIW tri-states the SIU_SERIRQ. The SIW drives the SIW_SERIRQ line low at the appropriate sample point if its associated IRQ/Data line is low, regardless of which device initiated the Start Frame.

The Sample Phase for each IRQ/Data follows the low to high transition of the Start Frame pulse by a number of clocks equal to the IRQ/Data Frame times three, minus one. (e.g., The IRQ5 Sample clock is the sixth IRQ/Data Frame, (6 x 3) - 1 = 17th clock after the rising edge of the Start Pulse).

Table 33-37. SIW_SERIRQ Sampling Periods (Sheet 1 of 2)

SIW_SERIRQ PERIOD	SIGNAL SAMPLED	# OF CLOCKS PAST START
1	Not Used	2
2	IRQ1	5
3	IRQ2	8
4	IRQ3	11
5	IRQ4	14
6	IRQ5	17
7	IRQ6	20
8	IRQ7	23
9	IRQ8	26
10	IRQ9	29
11	IRQ10	32

Table 33-37. SIW_SERIRQ Sampling Periods (Sheet 2 of 2)

SIW_SERIRQ PERIOD	SIGNAL SAMPLED	# OF CLOCKS PAST START
12	IRQ11	35
13	IRQ12	38
14	IRQ13	41
15	IRQ14	44
16	IRQ15	47

SIW_SERIRQ Period 14 is used to transfer IRQ13. Logical devices 4 (Serial Port 1), 5 (Serial Port 2) and 6 (WDT) shall have IRQ13 as a choice for their primary interrupt.

33.7.1.3 Stop Cycle Control

Once all IRQ/Data Frames have completed the Host Controller terminates SIW_SERIRQ activity by initiating a Stop Frame. Only the Host Controller can initiate the Stop Frame. A Stop Frame is indicated when the SIW_SERIRQ is low for two or three clocks. If the Stop Frame's low time is two clocks then the next SIW_SERIRQ Cycle's sampled mode is the Quiet mode; and any SIW_SERIRQ device may initiate a Start Frame in the second clock or more after the rising edge of the Stop Frame's pulse. If the Stop Frame's low time is three clocks then the next SIW_SERIRQ Cycle's sampled mode is the Continuous mode; and only the Host Controller may initiate a Start Frame in the second clock or more after the rising edge of the Stop Frame's pulse.

33.7.1.4 Latency

Latency for IRQ/Data updates over the SIW_SERIRQ bus in bridge-less systems with the minimum Host supported IRQ/Data Frames of seventeen, ranges up to 96 clocks (2.88 µs with a 33 MHz PCI Bus). If one or more PCI to PCI Bridge is added to a system, the latency for IRQ/ Data updates from the secondary or tertiary buses are a few clocks longer for synchronous buses, and approximately double for asynchronous buses.

33.7.1.5 EOI / ISR Read Latency

Any serialized IRQ scheme has a potential implementation issue related to IRQ latency. IRQ latency could cause an EOI or ISR Read to precede an IRQ transition that it should have followed. This could cause a system fault. The host interrupt controller is responsible for ensuring that these latency issues are mitigated. The recommended solution is to delay EOIs and ISR Reads to the interrupt controller by the same amount as the SIW_SERIRQ Cycle latency in order to ensure that these events do not occur out of order.

33.7.1.6 Reset and Initialization

The SIW_SERIRQ bus uses SIW_LRESET# as its reset signal. The SIW_SERIRQ pin is tri-stated by all agents while SIW_LRESET# is active. With reset, SIW_SERIRQ Slaves are put into the (continuous) IDLE mode. The Host Controller is responsible for starting the initial SIW_SERIRQ Cycle to collect system's IRQ/Data default values. The system then follows with the Continuous/ Quiet mode protocol (Stop Frame pulse width) for subsequent SIW_SERIRQ Cycles. It is Host Controller's responsibility to provide the default values to the Interrupt controller and other system logic before the first SIW_SERIRQ Cycle is performed. For SIW_SERIRQ system suspend, insertion, or removal application, the Host controller must be programmed into Continuous (IDLE) mode first. This is to guarantee SIW_SERIRQ bus is in IDLE state before the system configuration changes.

33.8 Configuration

The configuration of the SIW is very flexible and is based on the configuration architecture implemented in typical Plug-and-Play components. The SIW is designed for motherboard applications in which the resources required by their components are known. With its flexible resource allocation architecture, the SIW allows the BIOS to assign resources at POST.

33.8.1 Configuration Port Address

The SIW configuration port addresses for INDEX and DATA are controlled by the strap pin SIU1_DTR# during reset. When SIU1_DTR# is driven to '1' or left floating during reset SIW configuration port addresses are fixed at 4Eh/4Fh. When SIU1_DTR# is driven to '0' during reset SIW configuration port addresses are fixed at 20Eh/20Fh.

33.8.2 Primary Configuration Address Decoder

After a PCI Reset (SIW_LRESET# pin asserted) or Power On Reset the SIW is in the Run Mode with the UARTs and Watchdog timer disabled. They may be configured through two standard Configuration I/O Ports (INDEX and DATA) by placing the SIW into Configuration Mode.

The BIOS uses these configuration ports to initialize the logical devices at POST. The INDEX and DATA ports are only valid when the SIW is in Configuration Mode.

The INDEX and DATA ports are effective only when the chip is in the Configuration State. When the SIW is not in the Configuration State, reads return FFh and write data is ignored.

33.8.2.1 Entering the Configuration State

The device enters the Configuration State by the following contiguous sequence:

1. Write 80H to Configuration Index Port.
2. Write 86H to Configuration Index Port.

33.8.2.2 Exiting the Configuration State

The device exits the Configuration State by the following contiguous sequence:

1. Write 68H to Configuration Index Port.
2. Write 08H to Configuration Index Port.

33.8.2.3 Configuration Sequence

To program the configuration registers, the following sequence must be followed.

1. Enter Configuration Mode.
2. Configure the Configuration Registers.
3. Exit Configuration Mode.

33.8.2.4 Configuration Mode

The system sets the logical device information and activates desired logical devices through the INDEX and DATA ports. In configuration mode, the INDEX PORT is located at the CONFIG PORT address and the DATA PORT is at INDEX PORT address + 1.

The desired configuration registers are accessed in two steps:

1. Write the index of the Logical Device Number Configuration Register (i.e., 07) to the INDEX PORT and then write the number of the desired logical device to the DATA PORT.
2. Write the address of the desired configuration register within the logical device to the INDEX PORT and then write or read the configuration register through the DATA PORT.

Note: If accessing the Global Configuration Registers, step (1) is not required. The chip returns to the RUN State.

Note: Only two states are defined (Run and Configuration). In the Run State the chip is always ready to enter the Configuration State.

33.8.3 SIW Configuration Register Summary

The default values are defined with an h for hex, a bi for binary, or 00 for zero. If there is not a letter following the default value, assume it is a binary number.

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

Note: Reserved bits are Read Only.

Table 33-38. Configuration Register Summary (Sheet 1 of 2)

Global Configuration Registers			
Index	**Type**	**Default**	**Configuration Register**
07h	RW	00h	Logical Device Number
20h	R	00h	Device ID
21h	R	01h	Device Rev
28h	RW	01h	SIW I/F (wait states)
29h	RW	02h	SIRQ Configuration
2Eh	RW	00h	Test Mode Configuration Register
Logical Device 4 Registers (Serial Port 1)			
30h	RW	00h	Enable
60h	RW	00h	Base I/O Address MSB
61h	RW	00h	Base I/O Address LSB
70h	RW	00h	Primary Interrupt Select
74h	RW	04h	RSVD
75h	RW	04h	RSVD
F0h	RW	00h	RSVD
Logical Device 5 Registers (Serial Port 2)			
30h	RW	00h	Enable
60h	RW	00h	Base I/O Address MSB
61h	RW	00h	Base I/O Address LSB
70h	RW	00h	Primary Interrupt Select

Table 33-38. Configuration Register Summary (Sheet 2 of 2)

74h	RW	04h	RSVD
75h	RW	04h	RSVD
F0h	RW	00h	RSVD
Logical Device 6 Registers (Watchdog Timer)			
30h	RW	00h	Enable
60h	RW	00h	Base I/O Address MSB
61h	RW	00h	Base I/O Address LSB
70h	RW	00h	Primary Interrupt Select

33.8.3.1 Global Control/Configuration Registers [00h - 2Fh]

The chip-level (global) registers lie in the address range [00h-2Fh]. The design MUST use all eight bits of the ADDRESS Port for register selection. All unimplemented registers and bits ignore writes and return zero when read.

The INDEX PORT is used to select a configuration register in the chip. The DATA PORT is then used to access the selected register. These registers are accessible only in the Configuration Mode.

Register	Address (Type)	Description
Logical Device #` Default = 00h	07h (RW)	**Logical Device Select**: A write to this register selects the current logical device. This allows access to the control and configuration registers for each logical device.
Device ID Default = 00h	20h (R)	**Device ID:** A read only register which provides the Device ID.
Device Rev Default = 01h	21h (R)	**Device Rev:** A read only register which provides device revision information.
SIW Interface Default = 01h	28h c	Bit 7:1 RSVD = 0 Bit 0 LPC bus wait states 0 = Not Supported 1 = Long wait states (sync 6)
SIW Configuration Default = 02h	29h (RW - bit 0, 2, 3) (R - bit 1)	Bit 0 SIRQ enable 1 =enabled; enabled logical devices participate in interrupt generation 0 =disabled; serial interrupts disabled Bit 1 IRQ mode (READ ONLY, WRITES IGNORED) 1 =Continuous mode 0 =Quiet mode Bit 3:2 UART_CLK predivide 00 = divide by 1 01 = divide by 8 10 = divide by 26 11 = reserved Bit 7:4 RSVD = 0

Register	Address (Type)	Description
SIW Monitor Port Control Register Default = 00h	2D (RO - bit 7) (RW - bits 6:0)	Bit 0 UART1 Monitor Port Enable (UART1_MONPORTEN): Setting this bit enables the monitor port for UART1. This signal turns on all 8 UART2 monitor ports. Note: wdt_monporten, uart2_monporten and uart1_monporten must be set in a mutually exclusive manner i.e., only one monitor port enable must be set at one time. Bit 1 UART2 Monitor Port Enable (UART2_MONPORTEN): Setting this bit enables the monitor port for UART2. This signal turns on all 8 UART1 monitor ports. Note: wdt_monporten, uart2_monporten and uart1_monporten must be set in a mutually exclusive manner i.e., only one monitor port enable must be set at one time. Bit 2 WDT Monitor Port Enable (WDT_MONPORTEN): Setting this bit enables the monitor port for WDT. This signal turns on all 16 SIW monitor ports. Note: wdt_monporten, uart2_monporten and uart1_monporten must be set in a mutually exclusive manner i.e., only one monitor port enable must be set at one time. Bit 6:3 Monitor Port Slot Select[3:0] (MONPORTSEL): These bits select which Monitor Port Slot is enabled. These bits are used to select up to 16 slots within each UART1 and UART2 source group. Note: WDT has only 1 slot. So, port slot selection is not required. Bit 7 RSVD = 0
Default = 00h	2Eh	Reserved

33.8.3.2 Logical Device Configuration Registers [30h — FFh]

Used to access the registers that are assigned to each logical unit. This chip supports three logical units and has three sets of logical device registers. The three logical devices are UART1, UART2 and Watchdog Timer. A separate set (bank) of control and configuration registers exists for each logical device and is selected with the Logical Device # Register.

The INDEX PORT is used to select a specific logical device register. These registers are then accessed through the DATA PORT.

The Logical Device registers are accessible only when the device is in the Configuration State. The logical register addresses are shown in the tables below.

Table 33-39. Logical Device 4 (Serial Port 1)

Logical Device Register	Address	Description

Enable
Default = 00h
	30h
	(RW)
		Bits[7:1] Reserved, set to zero.
		Bit[0]
		1 =enable the logical device currently selected through the Logical Device # register.
		0 =Logical device currently selected is inactive

I/O Base Address Default = 00h	60h (RW) 61h (Bits 7:3 RW Bits 2:0 RO)	Registers 60h (MSB) and 61h (LSB) set the base address for the device. Note: Decode is on 8 Byte boundaries Comm Decode Ranges 3F8 - 3FF (COM 1) 2F8 - 2FF (COM 2) 220 - 227 228 - 22F 238 - 23F 2E8 - 2EF (COM 4) 338 - 33F 3E8 - 3EF (COM 3)
Primary Interrupt Select Default = 00h	70h (RW)	Bits[3:0] selects which interrupt level is used for the primary Interrupt. 00= no interrupt selected 01= IRQ1 02= IRQ2 03= IRQ3 04= IRQ4 05= IRQ5 06= IRQ6 07= IRQ7 08= IRQ8 09= IRQ9 0A= IRQ10 0B= IRQ11 0C= IRQ12 0D= IRQ13 0E= IRQ14 0F= IRQ15 Bits[7:4] Reserved Note: An Interrupt is activated by enabling this device (offset 30h), setting this register to a non-zero value and setting any combination of bits 0-4 in the corresponding UART IER and the occurrence of the corresponding UART event (i.e. Modem Status Change, Receiver Line Error Condition, Transmit Data Request, Receiver Data Available or Receiver Time Out) and setting the OUT2 bit in the MCR.
Reserved Default = 04h	74h	Bit 7:0 - Reserved
Reserved Default = 04h	75h	Bit 7:0 - Reserved
Reserved Default = 00h	F0h	Bit 7:0 - Reserved

Table 33-40. Logical Device 5 (Serial Port 2)

Logical Device Register	Address	Description

Enable
Default = 00h
	30h
	(RW)
		Bits[7:1] Reserved, set to zero.
		Bit[0]
		1 =enable the logical device currently selected through the Logical Device # register.
		0 =Logical device currently selected is inactive

I/O Base Address Default = 00h	60h (RW) 61h (Bits 7:3 RW Bits 2:0 RO)	Registers 60h (MSB) and 61h (LSB) set the base address for the device. Note: Decode is on 8 Byte boundaries Comm Decode Ranges 3F8 - 3FF (COM 1) 2F8 - 2FF (COM 2) 220 - 227 228 - 22F 238 - 23F 2E8 - 2EF (COM 4) 338 - 33F 3E8 - 3EF (COM 3)
Primary Interrupt Select Default = 00h	70h (RW)	Bits[3:0] selects which interrupt level is used for the primary Interrupt. 00= no interrupt selected 01= IRQ1 02= IRQ2 03= IRQ3 04= IRQ4 05= IRQ5 06= IRQ6 07= IRQ7 08= IRQ8 09= IRQ9 0A= IRQ10 0B= IRQ11 0C= IRQ12 0D= IRQ13 0E= IRQ14 0F= IRQ15 Bits[7:4] Reserved Note: An Interrupt is activated by enabling this device (offset 30h), setting this register to a non-zero value and setting any combination of bits 0-4 in the corresponding UART IER and the occurrence of the corresponding UART event (i.e. Modem Status Change, Receiver Line Error Condition, Transmit Data Request, Receiver Data Available or Receiver Time Out) and setting the OUT2 bit in the MCR.
Reserved Default = 04h	74h	Bit 7:0 - Reserved
Reserved Default = 04h	75h	Bit 7:0 - Reserved
Reserved Default = 00h	F0h	Bit 7:0 - Reserved

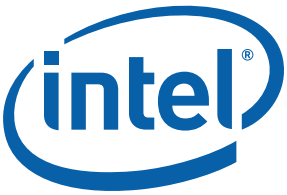
Table 33-41. Logical Device 6 (Watch Dog Timer)

Logical Device Register	Address	Description

Enable
Default = 00h
	30h
	(RW)
		Bits[7:1] Reserved, set to zero.
		Bit[0]
		1 =enable the logical device currently selected through the Logical Device # register.
		0 =Logical device currently selected is inactive

I/O Base Address Default = 00h	60h (RW) 61h (Bits 7:5 RW Bits 4:0 RO)	Registers 60h (MSB) and 61h (LSB) set the base address for the device. Note: Decode is on 32 Byte boundaries. WDT Base Address is generated by using the LPC Generic Decode Range 1 register (LG1), see D31:F0:84h for more details. This Base Address must be within the 128 bytes of LG1 Base register. Also the last byte accessed by the WDT must not exceed the LG1 Base Address +128 bytes.
Primary Interrupt Select Default = 00h	70h (RW)	Bits[3:0] selects which interrupt level is used for the primary Interrupt. 00= no interrupt selected 01= IRQ1 02= IRQ2 03= IRQ3 04= IRQ4 05= IRQ5 06= IRQ6 07= IRQ7 08= IRQ8 09= IRQ9 0A= IRQ10 0B= IRQ11 0C= IRQ12 0D= IRQ13 0E= IRQ14 0F= IRQ15 Bits[7:4] Reserved ***Note:*** An Interrupt is activated by enabling this device (offset 30h), setting this register to a non-zero value and when the first stage has been allowed to reach zero. An Interrupt is not generated if WDT_TOUT_CNF is set to change output after every timeout (See WDT Lock Register).

§ §



Acceleration and I/O Complex, Volume 4 of 6

Order Number: 320066-003US

34.0 PCI-to-PCI Bridge

34.1 Summary

The PCI-to-PCI bridge provides an interface between the AIOC and the MCH/IA. PCI-to-PCI Bridge forwards data and commands from AIOC memory target to the PCI-to-PCI bridge (Upstream interface). It also provide the IA a path into the AIOC via the PCI-to-PCI bridge (Downstream Interface).

34.2 PCI-to-PCI Bridge Detailed Register Descriptions

Table 34-1. Bus 0, Device 4, Function 0: Summary of PCI-to-PCI Bridge PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
0h	1h	"Offset 0h: VID: Vendor Identification Register" on page 1217	8086h
2h	3h	"Offset 2h: DID: Device Identification Register" on page 1217	5037h
4h	5h	"Offset 4h: PCICMD: Device Command Register" on page 1217	0h
6h	7h	"Offset 6h: PCISTS: PCI Device Status Register" on page 1218	10h
8h	8h	"Offset 8h: RID: Revision ID Register" on page 1219	Variable
9h	Bh	"Offset 9h: CC: Class Code Register" on page 1219	060400h
Ch	Ch	"Offset Ch: CLS: Cacheline Size Register" on page 1219	00h
Dh	Dh	"Offset Dh: LT: Latency Timer Register" on page 1220	00h
Eh	Eh	"Offset Eh: HDR: Header Type Register" on page 1220	1h
10h	14h	"Offset 10h: CSRBAR0: Control and Status Registers Base Address Register" on page 1220	00h
14h	17h	"Offset 14h: CSRBAR1: Control and Status Registers Base Address Register" on page 1221	00h
18h	18h	"Offset 18h: PBNUM: Primary Bus Number Register" on page 1221	00h
19h	19h	"Offset 19h: SECBNM: Secondary Bus Number Register" on page 1221	00h
1Ah	1Ah	"Offset 1Ah: SUBBNM: Subordinate Bus Number Register" on page 1222	00h
1Bh	1Bh	"Offset 1Bh: SECLT: Secondary Latency Timer Register" on page 1222	00h
1Ch	1Ch	"Offset 1Ch: IOB: I/O Base Register" on page 1222	F0
1Dh	1Dh	"Offset 1Dh: IOL: I/O Limit Register" on page 1223	0
1Eh	1Fh	"Offset 1Eh: SECSTA: Secondary Status Register" on page 1223	0h
20h	21h	"Offset 20h: MEMB: Memory Base Register" on page 1224	FFF0
22h	23h	"Offset 22h: MEML: Memory Limit Register" on page 1224	0
24h	25h	"Offset 24h: PMASE: Prefetchable Memory Base Register" on page 1225	FFF1H
26h	27h	"Offset 26h: PMLIMIT: Prefetchable Memory Limit Register" on page 1225	1H

Table 34-1. Bus 0, Device 4, Function 0: Summary of PCI-to-PCI Bridge PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
28h	28h	"Offset 28h: PMBASU: Memory Limit Register" on page 1226	Fh
2Ch	2Ch	"Offset 2Ch: PMLMTU: Prefetchable Memory Limit Upper Register" on page 1226	0
30h	31h	"Offset 30h: IOBU: I/O Base Upper Register" on page 1227	0
32h	33h	"Offset 32h: IOLU: I/O Limit Upper Register" on page 1227	0
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1227	dch
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1228	0
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1228	0
3Eh	3Fh	"Offset 3Eh: BCTL: Bridge Control Register" on page 1228	0000h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1229	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1230	00h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1230	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1231	0008h
E2h	E2h	"Offset E2h: PMCSE: Power Management Control and Status Extension Register" on page 1232	0000h

34.2.1 PCI-to-PCI Bridge Header

The PCI-to-PCI bridge header format is given in Table 34-2.

Table 34-2. PCI-to-PCI Bridge PCI Header

Byte Offset							
+7	+6	+5	+4	+3	+2	+1	+0
Status		Command		Device ID		Vendor ID	
BIST	Type	LT	CLS	Class Code			Rev
Base Address 1				Base Address 0			
Secondary Status		I/O Limit	I/O Base	Scdry. Lat. Timer	Subord. Bus Num	Scdry. Bus Num	Primary Bus Num
Pref. Mem. Limit		Pref. Mem. Base		Memory Limit		Memory Base	
Prefetchable Limit Upper 32 bits				Prefetchable Base Upper 32 bits			
Reserved			CP	I/O Limit Upper		I/O Base Upper	
Bridge Control		IRQ P	IRQ L	Expansion ROM Base			

34.2.2 PCI-to-PCI Bridge Configuration Space

34.2.2.1 Offset 0h: VID – Vendor Identification Register

Table 34-3. Offset 0h: VID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 0h **Offset End:** 1h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	Vendor ID		8086h	RO

34.2.2.2 Offset 2h: DID – Device Identification Register

Table 34-4. Offset 2h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 2h **Offset End:** 3h	
Size: 16 bit	**Default:** 5037h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	Device ID		5037h	RO

34.2.2.3 Offset 4h: PCICMD – Device Command Register

Table 34-5. Offset 4h: PCICMD: Device Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 4h **Offset End:** 5h	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	RV	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RW
07	RV	Reserved		0h	RV
06	PER	Parity Error Response		0h	RW
05	VPS	VGA Palette Snoop		0h	RO

Table 34-5. Offset 4h: PCICMD: Device Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 4h **Offset End:** 5h	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RW
01	MEM	Memory Space Enable		0h	RW
00	IO	I/O Space Enable		0h	RW

34.2.2.4 Offset 6h: PCISTS – Device Status Register

Table 34-6. Offset 6h: PCISTS: PCI Device Status Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 6h **Offset End:** 7h	
Size: 16 bit	**Default:** 10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	Detected Parity Error		0h	RO
14	SSE	Signaled System Error		0h	RO
13	RMA	Received Master Abort		0h	RO
12	RTA	Received Target Abort		0h	RO
11	STA	Signaled Target Abort		0h	RO
10 : 09	DST	DEVSEL Timing		00b	RO
08	MDPE	Master Data Parity Error		0h	RO
07	FB2B	Fast Back-to-Back Capable		0h	RO
06	RV	Reserved		0h	RV
05	MC66	66 MHz Capable		0h	RO
04	CL	Capabilities List		1	RO
03	IS	Interrupt Status		0h	RO
02 : 00	RV	Reserved		0h	RV

34.2.2.5 Offset 8h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 34-7. Offset 8h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 8h **Offset End:** 8h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	Revision ID. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

34.2.2.6 Offset 9h: CC – Class Code Register

Table 34-8. Offset 9h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 9h **Offset End:** Bh	
Size: 24 bit	**Default:** 060400h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	Class Code		060400h	RO

34.2.2.7 Offset Ch: CLS – Cacheline Size Register

Table 34-9. Offset Ch: CLS: Cacheline Size Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** Ch **Offset End:** Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CLS	Cacheline Size		0h	RO

34.2.2.8 Offset Dh: LT – Latency Timer Register

Table 34-10. Offset Dh: LT: Latency Timer Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:4:0	Offset Start: Dh Offset End: Dh	
Size: 8 bit	Default: 00h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
07 : 00	LT	Latency Timer		0h	RO

34.2.2.9 Offset Eh: HDR – Header Type Register

Table 34-11. Offset Eh: HDR: Header Type Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:4:0	Offset Start: Eh Offset End: Eh	
Size: 8 bit	Default: 1h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
07 : 00	HDR	PCI Header Type		1	RO

34.2.2.10 Offset 10h: CSRBAR0 – Control and Status Registers Base Address Register

Table 34-12. Offset 10h: CSRBAR0: Control and Status Registers Base Address Register

Description:					
View: PCI	BAR: Configuration		Bus:Device:Function: 0:4:0	Offset Start: 10h Offset End: 14h	
Size: 32 bit	Default: 00h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 04	ZERO	These bits of the address are hardwired to zero for 128 Kbyte.		0h	RO
03	PREF	Prefetchable		0h	RO
02 : 01	TYP	BAR Type (64-bit)		00b	RO
00	MEM	Memory Space Indicator		0h	RO

34.2.2.11 Offset 14h: CSRBAR1 – Control and Status Registers Base Address Register

Table 34-13. Offset 14h: CSRBAR1: Control and Status Registers Base Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	ZERO	These bits of the address are hardwired to zero for 128 Kbyte.		0h	RO
03	PREF	Prefetchable		0h	RO
02 : 01	TYP	BAR Type (64-bit)		0b	RO
00	MEM	Memory Space Indicator		0h	RO

34.2.2.12 Offset 18h: PBNUM – Primary Bus Number Register

Table 34-14. Offset 18h: PBNUM: Primary Bus Number Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 18h **Offset End:** 18h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PBNUM	Primary Bus Number		0h	RW

34.2.2.13 Offset 19h: SECBNM – Secondary Bus Number Register

Table 34-15. Offset 19h: SECBNM: Secondary Bus Number Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 19h **Offset End:** 19h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SECBNM	Secondary Bus Number		0h	RW

34.2.2.14 Offset 1Ah: SUBBNM – Subordinate Bus Number Register

Table 34-16. Offset 1Ah: SUBBNM: Subordinate Bus Number Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 1Ah **Offset End:** 1Ah	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SUBBNM	Subordinate Bus Number		0h	RW

34.2.2.15 Offset 1Bh: SECLT – Secondary Latency Timer Register

Table 34-17. Offset 1Bh: SECLT: Secondary Latency Timer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 1Bh **Offset End:** 1Bh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SECLT	Secondary Latency Timer		0h	RO

34.2.2.16 Offset 1Ch: IOB – I/O Base Register

This register (together with IOBU) specifies the starting I/O address of devices in the AIOC infrastructure. The range is aligned to a 4k boundary, so address bits [11:0] are assumed to be zero.

Table 34-18. Offset 1Ch: IOB: I/O Base Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 1Ch **Offset End:** 1Ch	
Size: 8 bit	**Default:** F0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	IOB	These bits correspond to address bits [15:12] of the I/O transaction. Address bits [13:16] are matched with IOBU.		Fh	RW
03 : 00	IOAW	The value “1” means that we implement 32-bit I/O space.		0	RO

34.2.2.17 Offset 1Dh: IOL – I/O Limit Register

This register (together with IOLU) specifies the ending I/O address of devices in the AIOC infrastructure. The range is aligned to a 4k boundary, so address bits [11:0] are assumed to be FFF.

Table 34-19. Offset 1Dh: IOL: I/O Limit Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 1Dh **Offset End:** 1Dh	
Size: 8 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	IOL	These bits correspond to address bits [15:12] of the I/O transaction. Address bits [13:16] are matched with IOLU.		0h	RW
03 : 00	IOAW	The value “1” means that we implement 32-bit I/O space.		0	RO

34.2.2.18 Offset 1Eh: SECSTA – Secondary Status Register

Table 34-20. Offset 1Eh: SECSTA: Secondary Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 1Eh **Offset End:** 1Fh	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	Detected Parity Error		0h	RO
14	RSE	Received System Error		0h	RO
13	RMA	Received Master Abort		0h	RO
12	RTA	Received Target Abort		0h	RO
11	STA	Signaled Target Abort		0h	RO
10 : 09	DST	DEVSEL Timing		00b	RO
08	MDPE	Master Data Parity Error		0h	RO
07	FB2B	Fast Back-to-Back Capable		0h	RO
06	RV	Reserved		0h	RV
05	MC66	66 MHz Capable		0h	RO
04 : 00	RV	Reserved		0h	RV

34.2.2.19 Offset 20h: MEMB – Memory Base Register

This register specifies the starting memory address of devices in the AIOC infrastructure. The range is aligned to a 1M boundary, so address bits [19:0] are assumed to be zero.

Table 34-21. Offset 20h: MEMB: Memory Base Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 20h **Offset End:** 21h	
Size: 16 bit	**Default:** FFF0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	MEMB	These bits correspond to address bits [31:20] of the transaction.		FFFh	RW
03 : 00	RV	Reserved		0h	RO

34.2.2.20 Offset 22h: MEML – Memory Limit Register

This register specifies the ending memory address of devices in the AIOC infrastructure. The range is aligned to a 1M boundary, so address bits [19:0] are assumed to be FFFFF.

Table 34-22. Offset 22h: MEML: Memory Limit Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 22h **Offset End:** 23h	
Size: 16 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	MEML	These bits correspond to address bits [31:20] of the transaction.		0h	RW
03 : 00	RV	Reserved		0h	RO

34.2.2.21 Offset 24h: PMBASE – Prefetchable Memory Base Register

Note: Prefetchable memory space is not used by AIOC in the EP80579. BIOS and enumeration software must be checked to make sure these default values are never modified to enable prefetchable memory space. Writing to this register can result in undefined behavior.

Table 34-23. Offset 24h: PMASE: Prefetchable Memory Base Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 24h **Offset End:** 25h	
Size: 16 bit	**Default:** FFF1H			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	PMBASE	These bits correspond to address bits [31:20] of the transaction.		FFFh	RW
03 : 01	MAMB	Memory Addressing Mode.		0h	RO
0	MBAUE	0 = Disabled 1 = Enabled. Base address is further defined by the bits of the memory base upper register		1	RO

34.2.2.22 Offset 26h: PMLIMIT – Prefetchable Memory Limit Register

Note: Prefetchable memory space is not used by AIOC in the EP80579. BIOS and enumeration software must be checked to make sure these default values are never modified to enable prefetchable memory space. Writing to this register can result in undefined behavior.

Table 34-24. Offset 26h: PMLIMIT: Prefetchable Memory Limit Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 26h **Offset End:** 27h	
Size: 16 bit	**Default:** 1H			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 04	MEML	These bits correspond to address bits [31:20] of the transaction.		0h	RW
03 : 01	MAML	Memory Addressing Mode.		0h	RO
0	MLAUE	0 = Disabled 1 = Enabled. Base address is further defined by the bits of the memory base upper register		1	RO

34.2.2.23 Offset 28h: PMBASU – Prefetchable Memory Base Upper Register

Table 34-25. Offset 28h: PMBASU: Memory Limit Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 28h **Offset End:** 28h	
Size: 8bit	**Default:** Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	RSV	These bits are reserved		0h	RO
03 : 00	BUA	Base Upper Address		Fh	RW

Note: Prefetchable memory space is not used by AIOC. BIOS and enumeration software must be checked to make sure these default values are never modified to enable prefetchable memory space. Writing to this register can result in undefined behavior.

34.2.2.24 Offset 2Ch: PMLMTU – Prefetchable Memory Limit Upper Register

Note: Prefetchable memory space is not used by AIOC. BIOS and enumeration software must be checked to make sure these default values are never modified to enable prefetchable memory space. Writing to this register can result in undefined behavior.

Table 34-26. Offset 2Ch: PMLMTU: Prefetchable Memory Limit Upper Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 2Ch **Offset End:** 2Ch	
Size: 8 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 04	RSV	Reserved		0h	RO
03 : 00	LUA	Limit Upper Address		0h	RW

34.2.2.25 Offset 30h: IOBU – I/O Base Upper Register

This register provides the upper 16 bits for IOB.

Table 34-27. Offset 30h: IOBU: I/O Base Upper Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 30h **Offset End:** 31h	
Size: 16 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	IOBU	These bits correspond to address bits [31:16] of the I/O transaction.		0h	RO

34.2.2.26 Offset 32h: IOLU – I/O Limit Upper Register

This register provides the upper 16 bits for IOL.

Table 34-28. Offset 32h: IOLU: I/O Limit Upper Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 32h **Offset End:** 33h	
Size: 16 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	IOLU	These bits correspond to address bits [31:16] of the I/O transaction.		0h	RO

34.2.2.27 Offset 34h: CP – Capabilities Pointer Register

Table 34-29. Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** dch			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	Capabilities Pointer		DCh	RO

34.2.2.28 Offset 3Ch: IRQL – Interrupt Line Register

Table 34-30. Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	Interrupt Line		0h	RO

34.2.2.29 Offset 3Dh: IRQP – Interrupt Pin Register

Table 34-31. Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 0			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	Interrupt Pin		1	RO

34.2.2.30 Offset 3Eh: BCTL – Bridge Control Register

Bits in this register such as VGA Enable and ISA Enable are RW bits for software compatibility, but don't affect the behavior of the bridge.

Table 34-32. Offset 3Eh: BCTL: Bridge Control Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 3Eh **Offset End:** 3Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 12	IRQP	Interrupt Pin		0	RO
11	DTSE	Discard Timer SERR Enable		0h	RO
10	DTS	Discard Timer Status		0h	RO
09	SDT	Secondary Discard Timer		0h	RO
08	PDT	Primary Discard Timer		0h	RO
07	FB2B	Fast Back to Back Enable		0h	RO
06	SECR	Secondary Bus Reset		0h	RW

Table 34-32. Offset 3Eh: BCTL: Bridge Control Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** 3Eh **Offset End:** 3Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	MAMODE	Master Abort Mode		0h	RO
04	VGA16	VGA 16-bit decode		0h	RO
03	VGAEN	VGA Enable		0h	RW
02	ISAEN	ISA Enable		0h	RW
01	SERREN	SERR# Enable		0h	RW
00	PREN	Parity Error Response Enable		0h	RW

34.2.2.31 Offset DCh: PCID – Power Management Capability ID Register

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 34-33. Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability)		01h	RO

34.2.2.32 Offset DDh: PCP – Power Management Next Capability Pointer Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 34-34. Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

34.2.2.33 Offset DEh: PMCAP – Power Management Capability Register

For an overview of the power management capability of the EP80579 integrated processor AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 34-35. Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** 0:4:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	Reserved	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

34.2.2.34 Offset E0h: PMCS – Power Management Control and Status Register

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 34-36. Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0008h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status - This bit is sticky for device 0 which is on suspend well, but not for the other 2 which aren’t on a power well.		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable - This bit is sticky for device 0 which is on suspend well, but not for the other 2 which aren’t on a power well.		0h	RO
07 : 04	Reserved	Reserved		0000b	RO
03	NSR	No Soft Reset		1	RO
02	Reserved	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

34.2.2.35 Offset E2h: PMCSE – Power Management Control and Status Extension Register

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 34-37. Offset E2h: PMCSE: Power Management Control and Status Extension Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** 0:4:0		**Offset Start:** E2h **Offset End:** E2h	
Size: 8 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
7	BPCC_EN	**Bus Power/Clock Control Enable)**- A “0” indicates that the bus power/clock control policies have been disabled. When the Bus Power/Clock Control mechanism is disabled, the bridge’s PMCSR **PowerState** field cannot be used by the system software to control the power or clock of the bridge’s secondary bus.		0h	RO
6	B2_B3_N	B2/3 Support. With BPCC_EN a ‘0’ this bit is a don’t care		0b	RO
5 :0	RSV	Reserved		0000b	RO

§ §

35.0 PCI-to-PCI Bridge: AIOC Configuration

35.1 Overview

The PCI-to-PCI bridge provides the IA-32 core/MCH fabric an interface to the AIOC. It implements PCI configuration registers that enable the IA-32 core to enumerate and configure the AIOC devices.

35.2 Feature List

The PCI-to-PCI bridge implements the following:

- Provides the IA-32 core/MCH complex access into the AIOC.

35.3 PCI Configuration Registers

The AIOC Type 1 PCI configuration registers can only be accessed using Type 1 Config Read/Write request types.

35.3.1 Description of PCI Configuration Header Space

The PCI configuration headers expose the various PCI devices on the AIOC to the IA infrastructure.

The PCI Specification requires implementation of 64 bytes of PCI Configuration registers as shown in Table 35-1. After a system reset, these registers are initially configured by the BIOS, and/or “Plug and Play” aware operating system. Device drivers will then read these registers to determine what resources (interrupt number, memory mapping location, etc.), the BIOS and/or OS assigned to the device.

Table 35-1. Type 0 PCI Configuration Header

Byte Offset							
+7	+6	+5	+4	+3	+2	+1	+0
Status		Command		Device ID		Vendor ID	
BIST	Type	LT	CLS	Class Code			Rev
Base Address 1				Base Address 0			
Base Address 3				Base Address 2			
Base Address 5				Base Address 4			
Subsys ID		Subsys ID		CardBus CIS Pointer			
Reserved			CP	Expansion ROM Base			
Mx L	Mn G	IRQ P	IRQ L	Reserved			

The fields shaded in grey are required in all PCI devices, and the other fields are optional. In each device section (below) the implemented fields are listed. An explanation of various PCI registers is provided below:

1. **Vendor ID** This uniquely identifies all Intel PCI products. A value of 0x8086 is the default for this field upon power up.
2. **Device ID** This uniquely identifies the Device.
3. **Command Reg**.Layout is shown below. Un-used bits are hard-wired to 0.

Figure 35-1. PCI Configuration Command Register Layout

Bit(s)	Initial Value	Description
0	0	I/O Access Enable
1	0	Memory Access Enable
2	0	Enable Mastering; Device in PCI-X mode is permitted to initiate a Split Completion transaction regardless of the state of this bit
3	0	Special Cycle Monitoring
4	0	Memory Write and Invalidate Enable, ignored by the device in PCI-X mode
5	0	Palette Snoop Enable
6	0	Parity Error Response
7	0	Wait Cycle Enable
8	0	SERR# Enable
9	0	Fast Back-to-Back Enable, ignored by the device in PCI-X mode
10	0	Interrupt Disable.
15:10	0	Reserved

4. **Status Register** Layout is shown below. Shaded bits are not used by this implementation and are hard–wired to 0.

Figure 35-2. PCI Configuration Status Register Layout

Bit(s)	Initial Value	Description
2:0	0	Reserved
3	0	Interrupt Status.
4	1	New Capabilities: Indicates that a device implements Extended Capabilities. A device sets this bit, and implements a capabilities list, to indicate that it supports PCI Power Management, PCI-X, and message signaled interrupts.
5	1	66MHz Capable
6	0	UDF Supported
7	0	Fast Back-to-Back Capable, This bit must be set to 0 in PCI-X mode.
8	0	Data Parity Reported
10:9	01	DEVSEL Timing (indicates medium device): Indicates conventional DEVSEL timing regardless of the operating mode.
11	0	Signaled Target Abort
12	0	Received Target Abort
13	0	Received Master Abort
14	0	Signaled System Error
15	0	Detected Parity Error

5. **Revision:** This is the 1st version of this device, so the revision number is 0x00.
6. **Class Code:** The class code, 0x020000 identifies this device as an Ethernet adapter.
7. **Cache Line Size**: Used to store the cache line size. The value is in units of 4 bytes. A system with a cache line size of 64B sets the value of this register to 0x10. The only sizes that are supported are 16, 32, 64, and 128 bytes. All other sizes are treated as 0. See exceptions in section 2.11.8 The default value at power up in 0x00.
8. Unsupported values affect PCI cache line support. All writes default to using the memory write (MW) command, and memory read command determination uses a cache-line size of 32 bytes.
9. **Latency Timer:** The lower 2 bits are not implemented and return 0. The upper 6 bits are RW. The default value of the Latency Timer register is 64 in PCI-X mode.
10. **Header Type**: This indicates if a device is single function or multifunction. The EP80579 returns a value of 0x00, indicating that it is a normal single function device.
11. **BIST:** BIT (Built in Self-test) will be not be implemented as supportable from PCI config space in this version of the device.
12. Base Address Register: The Base Address Registers (or BARs) are used to map the EP80579 register space to system memory space.

35.4 Interrupt Handling for AIOC Devices

Interrupts from the AIOC devices are routed towards the IA-32 core as either an MSI or by asserting the level-sensitive signal INTx.

The configuration provides a mechanism via the PCI configuration header space for functional units in the AIOC to interrupt the IA-32 core.

Most PCI configuration headers for AIOC devices implement two capability records to support interrupt and signal handling.

- A Message Signalled Interrupt capability record that follows the standard PCI format and describes the format of an IA MSI.
- An EP80579-specific Signal Target capability record that is unique to the EP80579 and describes how to target signals from the device.

The hardware will provide these capabilities to most of the devices implemented in the Configuration Bridge except for MDIO and the Local Expansion Bus. The software is responsible to enabling/disabling this functionality for each of the devices.

Table 35-2. Messaging and Signalling Capability Record per PCI Device (Sheet 1 of 2)

PCI Device	Function	Message Capable?	Signal Capable?	Interrupts (Max 8 per device)	Comment
0 (GbE0)	GbE	Yes	Yes	GBE0 Interrupt 0 GBE0 Interrupt 1 GBE0 Error Interrupt	
1 (GbE 1)	GbE	Yes	Yes	GBE1 Interrupt 0 GBE1 Interrupt 1 GBE1 Error Interrupt	
2 (GbE 2)	GbE	Yes	Yes	GBE2 Interrupt 0 GBE2 Interrupt 1 GBE2 Error Interrupt	

Table 35-2. Messaging and Signalling Capability Record per PCI Device (Sheet 2 of 2)

PCI Device	Function	Message Capable?	Signal Capable?	Interrupts (Max 8 per device)	Comment
3 (MDIO)	MDIO	No	No		
4 (CAN 0)	CAN	Yes	Yes	CAN0 System Interrupt CAN0 Parity Interrupt	
5 (CAN 1)	CAN	Yes	Yes	CAN1 System Interrupt CAN1 Parity Interrupt	
6 (SSP)	SSP	Yes	Yes	SSP Interrupt	
7 (1588)	1588 Interface	Yes	Yes	IEEE 1588 Interrupt	
8 (Local Expansion Bus)	Local Expansion Bus	Yes	Yes	EXP Expansion-Bus Parity Error Internal Interrupt	

35.5 Power Management of AIOC Devices

The AIOC devices support PCI Bus power management (PM). The supported states are:

- D0: Device is on and running which maximum power dissipation
- D3: Device is off and power can be removed. The device has lost its context information and will have be re-booted before it can enter the D0 state again.

The PM capabilities for the AIOC devices are defined as part of the capabilities list in the PCI Configuration space for each device. Figure 35-3 describes the contents of a power management entry in the capability list.

Figure 35-3. PCI Power Management Register Block

Byte 3	Byte 2	Byte 1	Byte 0
Power Management Capabilities (PMC)		Next Item Ptr	Capability ID
Data	PMCSR_BSE Bridge Support Extensions	Power Management Control / Status Register (PMCSR)	

PMCSR control bits are provided in part of PCI PM registers. These bits control the state of the device. The default value of this bit is 0 and corresponds to the D0 state.

Note: It is the responsibility of software to ensure that the devices and all of its function are in quiescent state

It is also the responsibility of software to disable Memory, I/O space (as appropriate) disabled, and interrupts (as appropriate) in the PCICMD register prior to powering down the device.

35.6 Gigabit Ethernet MAC Configuration Spaces: Bus M, Device 0-2, Function 0

Gigabit MAC 0, 1, and 2 are Devices 0, 1, and 2 of Bus M, respectively, and are accessed using type 1 configuration cycles. All MACs implement configuration spaces as defined in this section.

During an EEPROM read the configuration space will stall any configuration read or write cycles until after the EEPROM read has completed. For MEM/IO transfers it is up to the MAC to stall the transfer.

35.6.1 Register Details

Table 35-3. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1241	5040h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h

Table 35-3. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

Table 35-4. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1242	5044h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h

Table 35-4. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

Table 35-5. Bus M, Device2, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1241	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1242	5048h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1243	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1244	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1245	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1245	020000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1246	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1246	00000000h
14h	17h	"Offset 14h: IOBAR: CSR I/O Mapped BAR Register" on page 1247	00000001h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1248	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1248	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1249	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1249	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1250	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1251	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1251	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1252	X023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1253	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1254	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1254	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1255	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1255	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1256	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1257	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1258	05h

Table 35-5. Bus M, Device2, Function 0: Summary of Gigabit Ethernet MAC Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1258	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1259	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1259	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1260	0000h

35.6.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-6. Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 00h **Offset End:** 01h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 00h **Offset End:** 01h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel 8086h.		8086h	RO

35.6.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect. Each Gigabit MAC has its own unique DID.

Table 35-7. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5040h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 2	DIDH	**Device Identification Number High:** This is the upper 14-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #0.		1410h	RO
1 : 00	DIDL	**Device Identification Number Low:** This is the lower 2-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #0.		Fuse	RO

35.6.1.3 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect. Each Gigabit MAC has its own unique DID.

Table 35-8. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5044h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 2	DIDH	**Device Identification Number High:** This is the upper 14-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #1.		1411h	RO
1 : 00	DIDL	**Device Identification Number Low:** This is the lower 2-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #1.		Fuse	RO

35.6.1.4 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect. Each Gigabit MAC has its own unique DID.

Table 35-9. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5048h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 2	DIDH	**Device Identification Number High:** This is the upper 14-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #2.		1412h	RO
1 : 00	DIDL	**Device Identification Number Low:** This is the lower 2-bits of the full DID 16-bit value which is assigned to Gigabit Ethernet MAC #2.		Fuse	RO

35.6.1.5 Offset 04h: PCICMD – Device Command Register

Table 35-10. Offset 04h: PCICMD: Device Command Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 04h **Offset End:** 05h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 04h **Offset End:** 05h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RO
07	Reserved	Reserved		0h	RV
06	PER	Parity Error Response		0h	RO
05	VPS	VGA Palette Snoop		0h	RO
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RW
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** Setting this bit enables access to the I/O regions the device claims through its BARs.		0h	RW

35.6.1.6 Offset 06h: PCISTS – Device Status Register

Table 35-11. Offset 06h: PCISTS: PCI Device Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 06h **Offset End:** 07h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 06h **Offset End:** 07h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	Detected Parity Error: The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
14	SSE	Signaled System Error		0h	RO
13	RMA	Received Master Abort		0h	RO
12	RTA	Received Target Abort		0h	RO
11	STA	Signaled Target Abort		0h	RO
10 : 09	DST	DEVSEL Timing		00b	RO
08	MDPE	Master Data Parity Error: The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	FB2B	Fast Back-to-Back Capable		0h	RO
06	Reserved	Reserved		0h	RV
05	MC66	66 MHz Capable		0h	RO
04	CL	Capabilities List		1	RO
03	IS	Interrupt Status		0h	RO
02 : 00	Reserved	Reserved		0h	RV

35.6.1.7 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-12. Offset 08h: RID: Revision ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:0:0		**Offset Start:** 08h **Offset End:** 08h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:1:0		**Offset Start:** 08h **Offset End:** 08h	
View: PCI 3	**BAR:** Configuration	**Bus:Device:Function:** M:2:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.6.1.8 Offset 09h: CC – Class Code Register

Table 35-13. Offset 09h: CC: Class Code Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:0:0		**Offset Start:** 09h **Offset End:** 0Bh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:1:0		**Offset Start:** 09h **Offset End:** 0Bh	
View: PCI 3	**BAR:** Configuration	**Bus:Device:Function:** M:2:0		**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 020000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		020000h	RO

35.6.1.9 Offset 0Eh: HDR – Header Type Register

Table 35-14. Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.6.1.10 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the CSRs of a Gigabit Ethernet MAC. See Section 37.6, "GbE Controller Register Summary" for a description of the individual registers that this region exposes.

Table 35-15. Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 10h **Offset End:** 13h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 10h **Offset End:** 13h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 17	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
16 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 128KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.6.1.11 Offset 14h: IOBAR – CSR I/O Mapped BAR Register

The IOBAR is a PCI BAR in I/O space that allows access to the Gigabit MAC structures through IA I/O space. See Section 35.7, "Gigabit Ethernet MAC I/O Spaces: Bus M, Device 0-2, Function 0" on page 1261 for a description of the individual registers this region exposes.

Table 35-16. Offset 14h: IOBAR: CSR I/O Mapped BAR Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 14h **Offset End:** 17h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 14h **Offset End:** 17h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 00000001h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 05	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
04 : 02	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 32B.		0h	RO
01	Reserved	Reserved		0h	RO
00	TYP	**Addressing Type:** Hardwired to 1 to identify the region as in I/O space.		1	RO

Reads and writes to addresses mapped through this BAR are redirected to structures exposed through the CSRBAR (see Section 35.6.1.10, "Offset 10h: CSRBAR – Control and Status Registers Base Address Register" on page 1246).

35.6.1.12 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-17. Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.6.1.13 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-18. Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.6.1.14 Offset 34h: CP – Capabilities Pointer Register

The CP provides the offset to the location in configuration space where the first set of capabilities registers is located.

Table 35-19. Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 34h **Offset End:** 34h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 34h **Offset End:** 34h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.6.1.15 Offset 3Ch: IRQL – Interrupt Line Register

Table 35-20. Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	**Interrupt Line:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this device is connected to.		0h	RW

35.6.1.16 Offset 3Dh: IRQP – Interrupt Pin Register

Table 35-21. Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	**Interrupt Pin:** Set to 01h to indicate the device always uses INTA# as its interrupt pin.		01h	RO

35.6.1.17 Offset DCh: PCID – Power Management Capability ID Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-22. Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** DCh **Offset End:** DCh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** DCh **Offset End:** DCh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management)		01h	RO

35.6.1.18 Offset DDh: PCP – Power Management Next Capability Pointer Register

For an overview of the power management capability of EP80579 integrated processor AIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-23. Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** DDh **Offset End:** DDh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** DDh **Offset End:** DDh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** E4h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to E4h to indicate the offset of the next capability.		E4h	RO

35.6.1.19 Offset DEh: PMCAP – Power Management Capability Register

For an overview of the power management capability of AIOC devices, see Section 35.5, "Power Management of AIOC Devices". Note that Device 0 supports PME, Device 1 and 2 do not.

Table 35-24. Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** DEh **Offset End:** DFh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** DEh **Offset End:** DFh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** X023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		X	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	Reserved	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.6.1.20 Offset E0h: PMCS – Power Management Control and Status Register

For an overview of the power management capability of the EP80579 integrated processorAIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-25. Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** E0h **Offset End:** E1h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** E0h **Offset End:** E1h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status (sticky) This bit is sticky for device 0 which is on suspend well, but not for the other 2 which aren't on a power well. 0 = Writing a 0 has no effect. 1 = Set when GBE would normally assert the PME# signal independent of the state of the PME_En bit. ***Note:*** Writing a 1 to this bit will clear it and cause the internal PME to de-assert (if enabled). This bit must be explicitly cleared by the operating *Note:* On a write this register will be updated after a 100ns delay.	Y	0h	RWC
14 : 13	DATA_SCALE	Data Scale Hardwired to "00" because it does not support the associated Data register.		00b	RO
12 : 09	DATA_SEL	Data Select Hardwired to "0000" because it does not support the associated Data register.		0000b	RO
08	PME_EN	PME Enable (sticky) This bit is sticky for device 0 which is on suspend well, but not for the other 2 which aren't on a power well. A '1' enables GbE to generate an internal PME signal when PME_Status is '1'. This bit must be explicitly cleared by the operating system each time it is initially loaded. *Note:* On a write this register will be updated after a 100ns delay.	Y	0h	RW
07 : 04	Reserved	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	Reserved	Reserved		0h	RO
01 : 00	PS	Power State This 2-bit field is used both to determine the current power state of GbE function and to set a new power state. The definition of the field values are: 00b – D0 state 11b – $D3_{HOT}$ state If software attempts to write a value of 10b or 01b in to this field, the write operation must complete normally; however, the data is discarded and no state change occurs.		00b	RW

35.6.1.21 Offset E4h: SCID – Signal Target Capability ID Register

The Signal Target Capability record defines how the device targets it signals to IA agents. It is an 9B vendor-specific capability record and includes the SCID, SCP, SBC, STYP, SMIA, and SINT fields of the configuration header.

For more information on signaling by AIOC devices, see Section 35.4, "Interrupt Handling for AIOC Devices".

Table 35-26. Offset E4h: SCID: Signal Target Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:0:0		**Offset Start:** E4h **Offset End:** E4h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:1:0		**Offset Start:** E4h **Offset End:** E4h	
View: PCI 3	**BAR:** Configuration	**Bus:Device:Function:** M:2:0		**Offset Start:** E4h **Offset End:** E4h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCID	**Capability ID:** PCI SIG assigned capability record ID (09h, vendor specific)		09h	RO

35.6.1.22 Offset E5h: SCP – Signal Target Next Capability Pointer Register

Table 35-27. Offset E5h: SCP: Signal Target Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:0:0		**Offset Start:** E5h **Offset End:** E5h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:1:0		**Offset Start:** E5h **Offset End:** E5h	
View: PCI 3	**BAR:** Configuration	**Bus:Device:Function:** M:2:0		**Offset Start:** E5h **Offset End:** E5h	
Size: 8 bit	**Default:** F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCP	**Next Capability Pointer:** Hardwired to F0h to indicate the offset of the next capability.		F0h	RO

35.6.1.23 Offset E6h: SBC – Signal Target Byte Count Register

Table 35-28. Offset E6h: SBC: Signal Target Byte Count Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** E6h **Offset End:** E6h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** E6h **Offset End:** E6h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** E6h **Offset End:** E6h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBC	**Capability Record Byte Count:** Hardwired to the number of bytes in the vendor-specific capability record.		09h	RO

35.6.1.24 Offset E7h: STYP – Signal Target Capability Type Register

Table 35-29. Offset E7h: STYP: Signal Target Capability Type Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** E7h **Offset End:** E7h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** E7h **Offset End:** E7h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** E7h **Offset End:** E7h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	STYP	**Capability Record Type:** Vendor assigned capability record type (01h, EP80579 signal target capability)		01h	RO

35.6.1.25 Offset E8h: SMIA – Signal Target IA Mask Register

Table 35-30. Offset E8h: SMIA: Signal Target IA Mask Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** E8h **Offset End:** E8h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** E8h **Offset End:** E8h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** E8h **Offset End:** E8h	
Size: 8 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved			0h	RW
02	SMIA2	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect gbex_interrupt0 which carries all interrupt sources (functional plus AIOC internal bus errors and internal memory errors) and the functional interrupts are throttled by a timer		0h	RW
01	SMIA1	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect gbex_interrupt1 which carries all interrupt sources (functional plus AIOC internal bus errors and internal memory errors) but the functional interrupts are not throttled		0h	RW
00	SMIA0	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect gbex_error_interrupt which carries only AIOC internal bus errors and internal memory parity or uncorrectable ECC errors		0h	RW

35.6.1.26 Offset E9h: Reserved Register

Writing to this register will result in undefined behavior.

35.6.1.27 Offset EAh: Reserved Register

Writing to this register will result in undefined behavior.

35.6.1.28 Offset ECh: SINT – Signal Target Raw Interrupt Register

Table 35-31. Offset ECh: SINT: Signal Target Raw Interrupt Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** ECh **Offset End:** ECh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** ECh **Offset End:** ECh	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** ECh **Offset End:** ECh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 03	Reserved			0h	RO
02	SINT2	Interrupt: Read-only view of gbex_interrupt0 which carries all interrupt sources (functional plus AIOC internal bus errors and internal memory errors) and the functional interrupts are throttled by a timer		0h	RO
01	SINT1	Interrupt: Read-only view of gbex_interrupt1 which carries all interrupt sources (functional plus AIOC internal bus errors and internal memory errors) but the functional interrupts are not throttled		0h	RO
00	SINT0	Interrupt: Read-only view of gbex_error_interrupt which carries only AIOC internal bus errors and internal memory parity or uncorrectable ECC errors		0h	RO

35.6.1.29 Offset F0h: MCID – Message Signalled Interrupt Capability ID Register

The Message Signalled Interrupt Capability record defines how the device generates PCI MSI messages. It is an 10B PCI SIG-defined capability record and includes the MCID, MCP, MCTL, MADR, and MDATA fields of the configuration header.

Table 35-32. Offset F0h: MCID: Message Signalled Interrupt Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** F0h **Offset End:** F0h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** F0h **Offset End:** F0h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCID	**Capability ID:** PCI SIG assigned capability record ID (05h, MSI capability)		05h	RO

35.6.1.30 Offset F1h: MCP – Message Signalled Interrupt Next Capability Pointer Register

Table 35-33. Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** F1h **Offset End:** F1h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** F1h **Offset End:** F1h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** F1h **Offset End:** F1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.6.1.31 Offset F2h: MCTL – Message Signalled Interrupt Control Register

Table 35-34. Offset F2h: MCTL: Message Signalled Interrupt Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** F2h **Offset End:** F3h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** F2h **Offset End:** F3h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		0h	RO
08	MC	**Per-Vector Masking Capable:** Hardwired to 0 to indicate the device is not capable of per-vector masking.		0h	RO
07	C64	**64 bit Address Capable:** Hardwired to 0 to indicate the device does not generate 64b message addresses.		0h	RO
06 : 04	MME	**Multiple Message Enable:** System software writes to this field to indicate the number of allocated messages (less than or equal to the number of requested messages in MMC). A value of 0 corresponds to one message.		000h	RW
03 : 01	MMC	**Multiple Message Capable:** System software reads this field to determine the number of requested messages. Hardwired to 0 to request one message.		000h	RO
00	MSIE	**MSI Enable**: System software sets this bit to enable MSI signaling. A device driver is prohibited from writing this bit to mask a device's service request. If 1, the device can use an MSI to request service. If 0, the device cannot use an MSI to request service.		0h	RW

35.6.1.32 Offset F4h: MADR – Message Signalled Interrupt Address Register

Table 35-35. Offset F4h: MADR: Message Signalled Interrupt Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** F4h **Offset End:** F7h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** F4h **Offset End:** F7h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** F4h **Offset End:** F7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ADDR	**Message Address:** Written by the system to indicate the lower 32-bits of the address to use for the MSI memory write transaction. The lower two bits will always be written as 0.		0h	RW

35.6.1.33 Offset F8h: MDATA – Message Signalled Interrupt Data Register

Table 35-36. Offset F8h: MDATA: Message Signalled Interrupt Data Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:0:0	**Offset Start:** F8h **Offset End:** F9h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:1:0	**Offset Start:** F8h **Offset End:** F9h	
View: PCI 3	**BAR:** Configuration		**Bus:Device:Function:** M:2:0	**Offset Start:** F8h **Offset End:** F9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Message Data:** Written by the system to indicate the lower 16 bits of the data written in the MSI memory write DWORD transaction. The upper 16 bits of the transaction are written as 0.		0h	RW

35.7 Gigabit Ethernet MAC I/O Spaces: Bus M, Device 0-2, Function 0

The PCI-to-PCI Bridge implements IOADDR and IODATA registers in IA I/O space to allow access to Gigabit MAC structures through the IOBAR of a Gigabit MAC (see Section 35.6.1.11, “Offset 14h: IOBAR – CSR I/O Mapped BAR Register” on page 1247). These registers are used to indirectly access Gigabit MAC structures before the entire system memory map is available to use the MEMBAR. There are two registers for each MAC

During an EEPROM read the configuration space will stall any configuration read or write cycles until after the EEPROM read has completed. MEM/IO transfers generate an AIOC internal bus command and it is up to the Gig to stall the transfer.

35.7.1 Register Details

Table 35-37. Bus M, Device 0, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 35-38. Bus M, Device 1, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 35-39. Bus M, Device 2, Function 0: Gigabit Ethernet MAC I/O Spaces Registers

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"Offset 0000h: IOADDR - IOADDR Register" on page 1263	0000000h
0004h	0007h	"Offset 0004h: IODATA - IODATA Register" on page 1264	0000000h

Table 35-40. Gigabit Ethernet MAC I/O IOBAR Register Summary

Offset	Mnemonic	Name	RW	Size
00h - 03h	IOADDR	Internal register or internal memory address. 0000_0000h - 0001_FFFFh: Internal Registers/Memories 0002_0000h - FFFF_FFFFh: Undefined	RW	4 Byte
04h - 07h	IODATA	Data field for reads or writes to the Internal Register or Internal Memory as identified by the current value in IOADDR. All 32 bits of this register are read/write-able.	RW	4 Byte
08h - 3Fh	Reserved	Reserved	N/A	N/A

35.7.1.1 Offset 0000h: IOADDR - IOADDR Register

The IOADDR register specifies the structure within a GbE device that should be accessed on reads or writes to the IODATA register (see Section 35.7.1.2, "Offset 0004h: IODATA - IODATA Register" on page 1264).

Table 35-41. Offset 0000h: IOADDR - IOADDR Register

Description:					
View: PCI 1	**BAR:** IOBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 2	**BAR:** IOBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 3	**BAR:** IOBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0000h **Offset End:** 0003h	
Size: 32 bit	**Default:** 0000000h			**Power Well:** Vcc	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 17	Reserved	Reserved		0000h	RO
16 : 00	IOADDR	**Address for I/O Operation:** Provides the address for accesses to the GbE internal registers and memories allows access to full 128KB of space.		00000h	RW

The IOADDR register must always be written as a DWORD access. Writes that are less than 32 bits will be ignored. Reads of any size will return a DWORD of data. However, the chipset or CPU may only return a subset of that DWORD. IOADDR must be DWORD aligned.

Because only a particular range is addressable, the upper bits of this register are hard coded to zero. Bits 31 through 17 are not write-able and always read back as 0b.

At hardware reset, this register value resets to 00000000h. Once written, the value is retained until the next write or reset.

35.7.1.2 Offset 0004h: IODATA - IODATA Register

The IODATA register exposes the internal structures within the GbE identified by the last value written to the IOADDR register (see Section 35.7.1.1, "Offset 0000h: IOADDR - IOADDR Register" on page 1262) to read and write accesses.

Table 35-42. Offset 0004h: IODATA - IODATA Register

Description:					
View: PCI 1	**BAR:** IOBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0004h **Offset End:** 0007h	
View: PCI 2	**BAR:** IOBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0004h **Offset End:** 0007h	
View: PCI 3	**BAR:** IOBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0004h **Offset End:** 0007h	
Size: 32 bit	**Default:** 0000000h			**Power Well:** Vcc	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IODATA	**Data for I/O Operation:** Exposes the GbE device structure that IOADDR identifies to read and write accesses.		00000000h	RW

The IODATA register must always be written as a DWORD access when the IOADDR register contains a value for the Internal Registers and Memories. In this case, writes that are less than 32 bits will be ignored.

Reads to IODATA of any size will return a DWORD of data. However, the chipset or CPU may only return a subset of that DWORD.

Writes and reads to IODATA when the IOADDR register value is in an undefined range (0002_0000h -FFFF_FFFFh) should not be performed. Results are indeterministic.

Note: There are no special software timing requirements on accesses to IOADDR or IODATA. All accesses will be immediate except when data is not readily available or acceptable. In this case, the GbE will delay the results through normal bus methods (e.g., split transaction or NP Complete).

Note: Because a register/read or write takes two IO cycles to complete, software must provide a guarantee that the two IO cycles occur as an atomic operation. Otherwise, results can be non-deterministic from the software viewpoint.

35.8 GCU Configuration Space: Bus M, Device 3, Function 0

The GCU and MDIO port is Device 3 of Bus M, and is accessed using type 1 configuration cycles.

35.8.1 Register Details

Table 35-43. Bus M, Device 3, Function 0: Summary of GCU PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1265	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1266	503Eh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1266	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1267	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1268	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1268	FF0000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1268	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1269	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1269	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1270	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1270	DCh
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1270	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1271	00h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1271	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1272	0000h

35.8.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-44. Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel, 8086h.		8086h	RO

35.8.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-45. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 503Eh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the GCU device.		503E	RO

35.8.1.3 Offset 04h: PCICMD – Device Command Register

Table 35-46. Offset 04h: PCICMD: Device Command Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	**Interrupt Disable:** Setting this bit disables generation of interrupts by the GCU.		0h	RO
09	FBTB	**Fast Back-to-Back Enable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
08	SER	**SERR# Enable:** This bit is not implemented in the GCU and is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	Reserved	Reserved		0h	RV
06	PER	**Parity Error Enable:** This bit is not implemented in the GCU and is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
05	VPS	**VGA Palette Snoop Enable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
04	MWE	**Memory Write and Invalidate Enable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
03	SS	**Special Cycle Enable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
02	BM	**Bus Master Enable:** This bit is not implemented in the GCU and is hardwired to 0. GCU cannot be a bus master.		0h	RO
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** The device does not implement this functionality since it claims no I/O regions. The bit is hardwired to 0.		0h	RO

35.8.1.4 Offset 06h: PCISTS – Device Status Register

Table 35-47. Offset 06h: PCISTS: PCI Device Status Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
14	SSE	**Signaled System Error:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
13	RMA	**Received Master Abort Status:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
12	RTA	**Received Target Abort Status:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
11	STA	**Signaled Target Abort Status:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
10 : 09	DST	**DEVSEL Timing:** These bits are not implemented in the GCU and is hardwired to 0.		00b	RO
08	MDPE	**Master Data Parity Error Detected:** This bit is not implemented in the GCU and is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	FB2B	**Fast Back-to-Back Capable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
06	Reserved	Reserved		0h	RV
05	MC66	**66 MHz Capable:** This bit is not implemented in the GCU and is hardwired to 0.		0h	RO
04	CL	**Capabilities List:** This bit is hardwired to 1 to indicate that the device has a capabilities list.		1	RO
03	IS	**Interrupt Status:** This bit is not implemented in the GCU and is hardwired to 0. The GCU does not interrupt.		0h	RO
02 : 00	Reserved	Reserved		0h	RV

35.8.1.5 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-48. Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.8.1.6 Offset 09h: CC – Class Code Register

Table 35-49. Offset 09h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** FF0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		FF0000h	RO

35.8.1.7 Offset 0Eh: HDR – Header Type Register

Table 35-50. Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.8.1.8 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the GCU in the AIOC. See Section 38.4, "Register Summary" on page 1561 for a description of the individual registers that this region exposes.

Table 35-51. Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
11 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 4KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.8.1.9 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-52. Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.8.1.10 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-53. Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.8.1.11 Offset 34h: CP – Capabilities Pointer Register

The CP provides the offset to the location in configuration space where the first set of capabilities registers is located.

Table 35-54. Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.8.1.12 Offset DCh: PCID – Power Management Capability ID Register

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices”.

Table 35-55. Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability)		01h	RO

35.8.1.13 Offset DDh: PCP – Power Management Next Capability Pointer Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-56. Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.8.1.14 Offset DEh: PMCAP – Power Management Capability Register

For an overview of the power management capability of AIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-57. Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:3:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	Reserved	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.8.1.15 Offset E0h: PMCS – Power Management Control and Status Register

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices”.

Table 35-58. Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:3:0		**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status -		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable -		0h	RO
07 : 04	Reserved	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	Reserved	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

35.9 CAN Controller Configuration Spaces: Bus M, Device 4-5, Function 0

CAN controllers 0 and 1 are devices 4 and 5 of bus M, respectively and are accessed using type 1 configuration cycles. Both controllers implement configuration spaces as defined in this section.

35.9.1 Register Details

Table 35-59. Bus M, Device 4, Function 0: Summary of CAN Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1275	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1275	5039h
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1276	0h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1277	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1278	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1278	0C0900h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1279	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1279	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1280	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1280	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1281	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1281	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1282	01h
40h	40h	"Offset 40h: CANCTL - CAN Control Register" on page 1282	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1283	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1283	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1284	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1284	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1285	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1285	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1286	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1286	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1287	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1287	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1288	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1288	00h

Table 35-59. Bus M, Device 4, Function 0: Summary of CAN Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1289	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1289	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1290	0000h

Table 35-60. Bus M, Devices 5, Function 0: Summary of CAN Interface PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1275	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1276	503Ah
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1276	0h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1277	10h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1278	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1278	0C0900h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1279	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1279	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1280	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1280	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1281	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1281	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1282	01h
40h	40h	"Offset 40h: CANCTL - CAN Control Register" on page 1282	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1283	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1283	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1284	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1284	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1285	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1285	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1286	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1286	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1287	0h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1287	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1288	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1288	00h

Table 35-60. Bus M, Devices 5, Function 0: Summary of CAN Interface PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1289	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1289	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1290	0000h

35.9.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-61. Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 00h **Offset End:** 01h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel, 8086h.		8086h	RO

35.9.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Each CAN controller has its own device ID. Writes to this register have no effect.

Table 35-62. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 5039h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the CAN controller #1 device.		5039h	RO

35.9.1.3 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Each CAN controller has its own device ID. Writes to this register have no effect.

Table 35-63. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 503Ah			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the CAN controller #2 device.		503Ah	RO

35.9.1.4 Offset 04h: PCICMD – Device Command Register

Table 35-64. Offset 04h: PCICMD: Device Command Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 04h **Offset End:** 05h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RO
07	Reserved	Reserved		0h	RV
06	PER	Parity Error Response		0h	RO
05	VPS	VGA Palette Snoop		0h	RO
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RO
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** The device does not implement this functionality since it claims no I/O regions. The bit is hardwired to 0.		0h	RO

35.9.1.5 Offset 06h: PCISTS – Device Status Register

Table 35-65. Offset 06h: PCISTS: PCI Device Status Register

Description:						
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0		**Offset Start:** 06h **Offset End:** 07h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:**	10h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	Detected Parity Error: The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.			0h	RO
14	SSE	Signaled System Error			0h	RO
13	RMA	Received Master Abort			0h	RO
12	RTA	Received Target Abort			0h	RO
11	STA	Signaled Target Abort			0h	RO
10 : 09	DST	DEVSEL Timing			00b	RO
08	MDPE	Master Data Parity Error			0h	RO
07	FB2B	Fast Back-to-Back Capable			0h	RO
06	Reserved	Reserved			0h	RV
05	MC66	66 MHz Capable			0h	RO
04	CL	Capabilities List			1	RO
03	IS	Interrupt Status			0h	RO
02 : 00	Reserved	Reserved			0h	RV

35.9.1.6 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-66. Offset 08h: RID: Revision ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 08h **Offset End:** 08h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.9.1.7 Offset 09h: CC – Class Code Register

Table 35-67. Offset 09h: CC: Class Code Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 09h **Offset End:** 0Bh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 0C0900h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		0C0900h	RO

35.9.1.8 Offset 0Eh: HDR – Header Type Register

Table 35-68. Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 0Eh **Offset End:** 0Eh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.9.1.9 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the CAN controllers in the AIOC. See Section 39.6, “Register Summary” on page 1585 for a description of the individual registers that this region exposes.

Table 35-69. Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 10h **Offset End:** 13h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
11 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 4KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.9.1.10 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-70. Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 2Ch **Offset End:** 2Dh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.9.1.11 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-71. Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** 2Eh **Offset End:** 2Fh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.9.1.12 Offset 34h: CP – Capabilities Pointer Register

The CP provides the offset to the location in configuration space where the first set of capabilities registers is located.

Table 35-72. Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 34h **Offset End:** 34h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.9.1.13 Offset 3Ch: IRQL – Interrupt Line Register

Table 35-73. Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	**Interrupt Line:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this device is connected to.		0h	RW

35.9.1.14 Offset 3Dh: IRQP – Interrupt Pin Register

Table 35-74. Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	**Interrupt Pin:** Set to 01h to indicate the device always uses INTA# as its interrupt pin.		01h	RO

35.9.1.15 Offset 40h: CANCTL – CAN Control Register

Table 35-75. Offset 40h: CANCTL - CAN Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** 40h **Offset End:** 40h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** 40h **Offset End:** 40h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	RSVD	**Reserved**		00h	RO
00	Parity	**CAN RAM Parity**: 1 Even Parity for CAN SRAM, 0 Odd Parity.		0h	RW

35.9.1.16 Offset DCh: PCID – Power Management Capability ID Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 35-76. Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** DCh **Offset End:** DCh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability).		01h	RO

35.9.1.17 Offset DDh: PCP – Power Management Next Capability Pointer Register

Table 35-77. Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** DDh **Offset End:** DDh	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** E4h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to E4h to indicate the offset of the next capability.		E4h	RO

35.9.1.18 Offset DEh: PMCAP – Power Management Capability Register

Table 35-78. Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** DEh **Offset End:** DFh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	Reserved	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.9.1.19 Offset E0h: PMCS – Power Management Control and Status Register

Table 35-79. Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** E0h **Offset End:** E1h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status (sticky)		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable (sticky)		0h	RO
07 : 04	Reserved	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	Reserved	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

35.9.1.20 Offset E4h: SCID – Signal Target Capability ID Register

The Signal Target Capability record defines how the device targets it signals to IA agents. It is an 9B vendor-specific capability record and includes the SCID, SCP, SBC, STYP, SMIA, and SINT fields of the configuration header.

For more information on signaling by AIOC devices, see Section 35.4, "Interrupt Handling for AIOC Devices" on page 1235.

Table 35-80. Offset E4h: SCID: Signal Target Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** E4h **Offset End:** E4h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** E4h **Offset End:** E4h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCID	**Capability ID:** PCI SIG assigned capability record ID (09h, vendor specific)		09h	RO

35.9.1.21 Offset E5h: SCP – Signal Target Next Capability Pointer Register

Table 35-81. Offset E5h: SCP: Signal Target Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** E5h **Offset End:** E5h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** E5h **Offset End:** E5h	
Size: 8 bit	**Default:** F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCP	**Next Capability Pointer:** Hardwired to F0h to indicate the offset of the next capability.		F0h	RO

35.9.1.22 Offset E6h: SBC – Signal Target Byte Count Register

Table 35-82. Offset E6h: SBC: Signal Target Byte Count Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** E6h **Offset End:** E6h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** E6h **Offset End:** E6h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBC	**Capability Record Byte Count:** Hardwired to the number of bytes in the vendor-specific capability record.		09h	RO

35.9.1.23 Offset E7h: STYP – Signal Target Capability Type Register

Table 35-83. Offset E7h: STYP: Signal Target Capability Type Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** E7h **Offset End:** E7h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** E7h **Offset End:** E7h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	STYP	**Capability Record Type:** Vendor assigned capability record type (01h, EP80579 signal target capability).		01h	RO

35.9.1.24 Offset E8h: SMIA – Signal Target IA Mask Register

Table 35-84. Offset E8h: SMIA: Signal Target IA Mask Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** E8h **Offset End:** E8h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** E8h **Offset End:** E8h	
Size: 8 bit	**Default:** 0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved			0h	RW
01	SMIA1	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect CANx Parity Interrupt		0h	RW
00	SMIA0	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect CANx System Interrupt.		0h	RW

35.9.1.25 Offset E9h: Reserved Register

Writing to this register will result in undefined behavior

35.9.1.26 Offset EAh: Reserved Register

Writing to this register will result in undefined behavior.

35.9.1.27 Offset ECh: SINT – Signal Target Raw Interrupt Register

Table 35-85. Offset ECh: SINT: Signal Target Raw Interrupt Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** ECh **Offset End:** ECh	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** ECh **Offset End:** ECh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved			0h	RO
01	SINT1	Interrupt: Read-only view of CANx Parity Interrupt		0h	RO
00	SINT0	Interrupt: Read-only view of CANx System Interrupt		0h	RO

35.9.1.28 Offset F0h: MCID – Message Signalled Interrupt Capability ID Register

The Message Signalled Interrupt Capability record defines how the device generates PCI MSI messages. It is an 10B PCI SIG-defined capability record and includes the MCID, MCP, MCTL, MADR, and MDATA fields of the configuration header.

Table 35-86. Offset F0h: MCID: Message Signalled Interrupt Capability ID Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** F0h **Offset End:** F0h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCID	**Capability ID:** PCI SIG assigned capability record ID (05h, MSI capability)		05h	RO

35.9.1.29 Offset F1h: MCP – Message Signalled Interrupt Next Capability Pointer Register

Table 35-87. Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** F1h **Offset End:** F1h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** F1h **Offset End:** F1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.9.1.30 Offset F2h: MCTL – Message Signalled Interrupt Control Register

Table 35-88. Offset F2h: MCTL: Message Signalled Interrupt Control Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** F2h **Offset End:** F3h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		0h	RO
08	MC	**Per-Vector Masking Capable:** Hardwired to 0 to indicate the device is not capable of per-vector masking.		0h	RO
07	C64	**64 bit Address Capable:** Hardwired to 0 to indicate the device does not generate 64b message addresses.		0h	RO
06 : 04	MME	**Multiple Message Enable:** System software writes to this field to indicate the number of allocated messages (less than or equal to the number of requested messages in MMC). A value of 0 corresponds to one message.		000h	RW
03 : 01	MMC	**Multiple Message Capable:** System software reads this field to determine the number of requested messages. Hardwired to 0 to request one message.		000h	RO
00	MSIE	MSI Enable: System software sets this bit to enable MSI signaling. A device driver is prohibited from writing this bit to mask a device's service request. If 1, the device can use an MSI to request service. If 0, the device cannot use an MSI to request service.		0h	RW

35.9.1.31 Offset F4h: MADR – Message Signalled Interrupt Address Register

Table 35-89. Offset F4h: MADR: Message Signalled Interrupt Address Register

Description:					
View: PCI 1	**BAR:** Configuration		**Bus:Device:Function:** M:4:0	**Offset Start:** F4h **Offset End:** F7h	
View: PCI 2	**BAR:** Configuration		**Bus:Device:Function:** M:5:0	**Offset Start:** F4h **Offset End:** F7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ADDR	**Message Address:** Written by the system to indicate the lower 32-bits of the address to use for the MSI memory write transaction. The lower two bits will always be written as 0.		0h	RW

35.9.1.32 Offset F8h: MDATA – Message Signalled Interrupt Data Register

Table 35-90. Offset F8h: MDATA: Message Signalled Interrupt Data Register

Description:					
View: PCI 1	**BAR:** Configuration	**Bus:Device:Function:** M:4:0		**Offset Start:** F8h **Offset End:** F9h	
View: PCI 2	**BAR:** Configuration	**Bus:Device:Function:** M:5:0		**Offset Start:** F8h **Offset End:** F9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Message Data:** Written by the system to indicate the lower 16 bits of the data written in the MSI memory write DWORD transaction. The upper 16 bits of the transaction are written as 0.		0h	RW

35.10 SSP Controller Configuration Space: Bus M, Device 6, Function 0

The SSP controller is Device 6 of Bus M, and is accessed using type 1 configuration cycles

35.10.1 Register Details

Table 35-91. Bus M, Device 6, Function 0: Summary of SSP Controller PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1292	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1292	503Bh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1292	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1293	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1294	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1295	078000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1295	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1295	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1296	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1296	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1297	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1297	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1297	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1298	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1298	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1298	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1299	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1300	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1300	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1300	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1301	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1301	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1302	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1302	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1302	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1303	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1303	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1304	0000h

35.10.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-92. Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel, 8086h.		8086h	RO

35.10.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-93. Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 503Bh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the SSP Controller device.		503Bh	RO

35.10.1.3 Offset 04h: PCICMD – Device Command Register

Table 35-94. Offset 04h: PCICMD: Device Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RO

Table 35-94. Offset 04h: PCICMD: Device Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0h	RV
06	PER	Parity Error Response		0h	RO
05	VPS	VGA Palette Snoop		0h	RO
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RO
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** The device does not implement this functionality since it claims no I/O regions. The bit is hardwired to 0.		0h	RO

35.10.1.4 Offset 06h: PCISTS – Device Status Register

Table 35-95. Offset 06h: PCISTS: PCI Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
14	SSE	**Signaled System Error:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
13	RMA	**Received Master Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
12	RTA	**Received Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
11	STA	**Signaled Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
10 : 09	DST	**DEVSEL Timing:** The device does not implement this functionality. These bits are hardwired to 0.		00b	RO
08	MDPE	**Master Data Parity Error Detected:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	FB2B	**Fast Back-to-Back Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
06	Reserved	Reserved		0h	RV

Table 35-95. Offset 06h: PCISTS: PCI Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	MC66	**66 MHz Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
04	CL	**Capabilities List:** This bit is hardwired to 1 to indicate that the device has a capabilities list.		1	RO
03	IS	**Interrupt Status:**		0h	RO
02 : 00	Reserved	Reserved		0h	RV

35.10.1.5 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-96. Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.10.1.6 Offset 09h: CC – Class Code Register

Table 35-97. Offset 09h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 078000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		078000h	RO

35.10.1.7 Offset 0Eh: HDR – Header Type Register

Table 35-98. Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.10.1.8 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the control and status registers for the SSP Unit. See Section 40.4, “Register Summary” on page 1606 for a description of the registers this region exposes.

Table 35-99. Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
11 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 4KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.10.1.9 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-100. Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.10.1.10 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset,

Table 35-101.Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.10.1.11 Offset 34h: CP – Capabilities Pointer Register

Table 35-102.Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.10.1.12 Offset 3Ch: IRQL – Interrupt Line Register

Table 35-103.Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	**Interrupt Line:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this device is connected to.		0h	RW

35.10.1.13 Offset 3Dh: IRQP – Interrupt Pin Register

Table 35-104.Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	**Interrupt Pin:** Set to 01h to indicate the device always uses INTA# as its interrupt pin.		01h	RO

35.10.1.14 Offset DCh: PCID – Power Management Capability ID Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 35-105.Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability)		01h	RO

35.10.1.15 Offset DDh: PCP – Power Management Next Capability Pointer Register

Table 35-106.Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** E4h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to E8h to indicate the offset of the next capability.		E4h	RO

35.10.1.16 Offset DEh: PMCAP - Power Management Capability

Table 35-107.Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	RV	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.10.1.17 Offset E0h: PMCS – Power Management Control and Status Register

Table 35-108.Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status (sticky)		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable (sticky)		0b	RO
07 : 04	RV	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	RV	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

35.10.1.18 Offset E4h: SCID – Signal Target Capability ID Register

The Signal Target Capability record defines how the device targets it signals to IA agents. It is an 9B vendor-specific capability record and includes the SCID, SCP, SBC, STYP, SMIA, and SINT fields of the configuration header.

For more information on signaling by AIOC devices, see Section 35.4, “Interrupt Handling for AIOC Devices” on page 1235.

Table 35-109.Offset E4h: SCID: Signal Target Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** E4h **Offset End:** E4h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCID	**Capability ID:** PCI SIG assigned capability record ID (09h, vendor specific)		09h	RO

35.10.1.19 Offset E5h: SCP – Signal Target Next Capability Pointer Register

Table 35-110.Offset E5h: SCP: Signal Target Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** E5h **Offset End:** E5h	
Size: 8 bit	**Default:** F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCP	**Next Capability Pointer:** Hardwired to F0h to indicate the offset of the next capability.		F0h	RO

35.10.1.20 Offset E6h: SBC – Signal Target Byte Count Register

Table 35-111.Offset E6h: SBC: Signal Target Byte Count Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** E6h **Offset End:** E6h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBC	**Capability Record Byte Count:** Hardwired to the number of bytes in the vendor-specific capability record.		09h	RO

35.10.1.21 Offset E7h: STYP – Signal Target Capability Type Register

Table 35-112.Offset E7h: STYP: Signal Target Capability Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** E7h **Offset End:** E7h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	STYP	**Capability Record Type:** Vendor assigned capability record type (01h, EP80579 signal target capability)		01h	RO

35.10.1.22 Offset E8h: SMIA – Signal Target IA Mask Register

Table 35-113.Offset E8h: SMIA: Signal Target IA Mask Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** E8h **Offset End:** E8h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved			0h	RW
00	SMIA	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect SSP Interrupt		0h	RW

35.10.1.23 Offset E9h: Reserved Register

Writing to the register will result in undefined behavior.

35.10.1.24 Offset EAh: Reserved Register

Writing to this register will result in undefined behavior.

35.10.1.25 Offset ECh: SINT – Signal Target Raw Interrupt Register

Table 35-114.Offset ECh: SINT: Signal Target Raw Interrupt Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** ECh **Offset End:** ECh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved			0h	RO
00	SINT	Interrupt: Read-only view of SSP Interrupt		0h	RO

35.10.1.26 Offset F0h: MCID – Message Signalled Interrupt Capability ID Register

The Message Signalled Interrupt Capability record defines how the device generates PCI MSI messages. It is an 10B PCI SIG-defined capability record and includes the MCID, MCP, MCTL, MADR, and MDATA fields of the configuration header.

Table 35-115.Offset F0h: MCID: Message Signalled Interrupt Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCID	**Capability ID:** PCI SIG assigned capability record ID (05h, MSI capability)		05h	RO

35.10.1.27 Offset F1h: MCP – Message Signalled Interrupt Next Capability Pointer Register

Table 35-116.Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** F1h **Offset End:** F1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.10.1.28 Offset F2h: MCTL – Message Signalled Interrupt Control Register

Table 35-117.Offset F2h: MCTL: Message Signalled Interrupt Control Register (Sheet 1 of

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:6:0	**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		0h	RO
08	MC	**Per-Vector Masking Capable:** Hardwired to 0 to indicate the device is not capable of per-vector masking.		0h	RO
07	C64	**64 bit Address Capable:** Hardwired to 0 to indicate the device does not generate 64b message addresses.		0h	RO

Table 35-117.Offset F2h: MCTL: Message Signalled Interrupt Control Register (Sheet 2 of

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
06 : 04	MME	**Multiple Message Enable:** System software writes to this field to indicate the number of allocated messages (less than or equal to the number of requested messages in MMC). A value of 0 corresponds to one message.		000h	RW
03 : 01	MMC	**Multiple Message Capable:** System software reads this field to determine the number of requested messages. Hardwired to 0 to request one message.		000h	RO
00	MSIE	MSI Enable: System software sets this bit to enable MSI signaling. A device driver is prohibited from writing this bit to mask a device's service request. If 1, the device can use an MSI to request service. If 0, the device cannot use an MSI to request service.		0h	RW

35.10.1.29 Offset F4h: MADR – Message Signalled Interrupt Address Register

Table 35-118.Offset F4h: MADR: Message Signalled Interrupt Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** F4h **Offset End:** F7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ADDR	**Message Address:** Written by the system to indicate the lower 32-bits of the address to use for the MSI memory write transaction. The lower two bits will always be written as 0.		0h	RW

35.10.1.30 Offset F8h: MDATA – Message Signalled Interrupt Data Register

Table 35-119.Offset F8h: MDATA: Message Signalled Interrupt Data Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:6:0		**Offset Start:** F8h **Offset End:** F9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Message Data:** Written by the system to indicate the lower 16 bits of the data written in the MSI memory write DWORD transaction. The upper 16 bits of the transaction are written as 0.		0h	RW

35.11 IEEE 1588 Hardware Assist Unit Configuration Space: Bus M, Device 7, Function 0

The IEEE 1588 Hardware Assist Unit (Device 7 of Bus M) is accessed using type 1 configuration cycles.

35.11.1 Register Details

Table 35-120.Bus M, Device 7, Function 0: Summary of IEEE 1588 Timestamp Unit PCI Configuration Registers

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1306	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1306	503Ch
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1306	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1307	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1308	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1308	111000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1309	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1309	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1310	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1310	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1310	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1311	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1311	01h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1312	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1312	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1313	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1313	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1314	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1314	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1314	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1315	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1315	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1316	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1316	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1317	00h
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1317	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1318	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1318	0000h

35.11.1.1 Offset 00h: VID – Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-121.Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel, 8086h.		8086h	RO

35.11.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-122.Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 503Ch			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the IEEE 1588 Hardware Assist Unit.		503Ch	RO

35.11.1.3 Offset 04h: PCICMD – Device Command Register

Table 35-123.Offset 04h: PCICMD: Device Command Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RO

Table 35-123.Offset 04h: PCICMD: Device Command Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	Reserved	Reserved		0h	RV
06	PER	Parity Error Response		0h	RO
05	VPS	VGA Palette Snoop		0h	RO
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RO
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** The device does not implement this functionality since it claims no I/O regions. The bit is hardwired to 0.		0h	RO

35.11.1.4 Offset 06h: PCISTS – Device Status Register

Table 35-124.Offset 06h: PCISTS: PCI Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
14	SSE	**Signaled System Error:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
13	RMA	**Received Master Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
12	RTA	**Received Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
11	STA	**Signaled Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
10 : 09	DST	**DEVSEL Timing:** The device does not implement this functionality. These bits are hardwired to 0.		00b	RO
08	MDPE	**Master Data Parity Error Detected:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	FB2B	**Fast Back-to-Back Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
06	Reserved	Reserved		0h	RV

Table 35-124.Offset 06h: PCISTS: PCI Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
05	MC66	**66 MHz Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
04	CL	**Capabilities List:** This bit is hardwired to 1 to indicate that the device has a capabilities list.		1	RO
03	IS	**Interrupt Status:**		0h	RO
02 : 00	Reserved	Reserved		0h	RV

35.11.1.5 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-125.Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.11.1.6 Offset 09h: CC – Class Code Register

Table 35-126.Offset 09h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 111000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		111000h	RO

35.11.1.7 Offset 0Eh: HDR – Header Type Register

Table 35-127.Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.11.1.8 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the control and status registers for the 1588 Hardware Assist Unit. See Section 41.6, “Register Summary” on page 1637 for a description of the registers this region exposes.

Table 35-128.Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
11 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 4KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.11.1.9 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-129.Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.11.1.10 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-130.Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.11.1.11 Offset 34h: CP – Capabilities Pointer Register

Table 35-131.Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.11.1.12 Offset 3Ch: IRQL – Interrupt Line Register

Table 35-132.Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	**Interrupt Line:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this device is connected to.		0h	RW

35.11.1.13 Offset 3Dh: IRQP – Interrupt Pin Register

Table 35-133.Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	**Interrupt Pin:** Set to 01h to indicate the device always uses INTA# as its interrupt pin.		01h	RO

35.11.1.14 Offset DCh: PCID – Power Management Capability ID Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, “Power Management of AIOC Devices” on page 1236.

Table 35-134.Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability)		01h	RO

35.11.1.15 Offset DDh: PCP – Power Management Next Capability Pointer Register

Table 35-135.Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** E4h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to E8h to indicate the offset of the next capability.		E4h	RO

35.11.1.16 Offset DEh: PMCAP – Power Management Capability Register

Table 35-136.Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	RV	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.11.1.17 Offset E0h: PMCS – Power Management Control and Status Register

Table 35-137.Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status (sticky)		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable (sticky)		0h	RO
07 : 04	RV	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	RV	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

35.11.1.18 Offset E4h: SCID – Signal Target Capability ID Register

The Signal Target Capability record defines how the device targets it signals to IA agents. It is an 9B vendor-specific capability record and includes the SCID, SCP, SBC, STYP, SMIA, and SINT fields of the configuration header.

For more information on signaling by AIOC devices, see Section 35.4, “Interrupt Handling for AIOC Devices” on page 1235.

Table 35-138.Offset E4h: SCID: Signal Target Capability ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** E4h **Offset End:** E4h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCID	**Capability ID:** PCI SIG assigned capability record ID (09h, vendor specific)		09h	RO

35.11.1.19 Offset E5h: SCP – Signal Target Next Capability Pointer Register

Table 35-139.Offset E5h: SCP: Signal Target Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** E5h **Offset End:** E5h	
Size: 8 bit	**Default:** F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCP	**Next Capability Pointer:** Hardwired to F0h to indicate the offset of the next capability.		F0h	RO

35.11.1.20 Offset E6h: SBC – Signal Target Byte Count Register

Table 35-140.Offset E6h: SBC: Signal Target Byte Count Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:7:0		**Offset Start:** E6h **Offset End:** E6h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBC	**Capability Record Byte Count:** Hardwired to the number of bytes in the vendor-specific capability record.		09h	RO

35.11.1.21 Offset E7h: STYP – Signal Target Capability Type Register

Table 35-141.Offset E7h: STYP: Signal Target Capability Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** E7h **Offset End:** E7h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	STYP	**Capability Record Type:** Vendor assigned capability record type (01h, EP80579 signal target capability)		01h	RO

35.11.1.22 Offset E8h: SMIA – Signal Target IA Mask Register

Table 35-142.Offset E8h: SMIA: Signal Target IA Mask Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** E8h **Offset End:** E8h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved			0h	RW
00	SMIA	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect IEEE 1588 Interrupt		0h	RW

35.11.1.23 Offset E9h: Reserved Register

Writing to this register will result in undefined behavior.

35.11.1.24 Offset EAh: Reserved Register

Writing to this register will result in undefined behavior.

35.11.1.25 Offset ECh: SINT – Signal Target Raw Interrupt Register

Table 35-143.Offset ECh: SINT: Signal Target Raw Interrupt Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** ECh **Offset End:** ECh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 01	Reserved			0h	RO
00	SINT	Interrupt: Read-only view of IEEE 1588 Interrupt		0h	RO

35.11.1.26 Offset F0h: MCID – Message Signalled Interrupt Capability ID Register

The Message Signalled Interrupt Capability record defines how the device generates PCI MSI messages. It is an 10B PCI SIG-defined capability record and includes the MCID, MCP, MCTL, MADR, and MDATA fields of the configuration header.

Table 35-144.Offset F0h: MCID: Message Signalled Interrupt Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCID	**Capability ID:** PCI SIG assigned capability record ID (05h, MSI capability)		05h	RO

35.11.1.27 Offset F1h: MCP – Message Signalled Interrupt Next Capability Pointer Register

Table 35-145.Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** F1h **Offset End:** F1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.11.1.28 Offset F2h: MCTL – Message Signalled Interrupt Control Register

Table 35-146.Offset F2h: MCTL: Message Signalled Interrupt Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		0h	RO
08	MC	**Per-Vector Masking Capable:** Hardwired to 0 to indicate the device is not capable of per-vector masking.		0h	RO
07	C64	**64 bit Address Capable:** Hardwired to 0 to indicate the device does not generate 64b message addresses.		0h	RO
06 : 04	MME	**Multiple Message Enable:** System software writes to this field to indicate the number of allocated messages (less than or equal to the number of requested messages in MMC). A value of 0 corresponds to one message.		000h	RW
03 : 01	MMC	**Multiple Message Capable:** System software reads this field to determine the number of requested messages. Hardwired to 0 to request one message.		000h	RO
00	MSIE	MSI Enable: System software sets this bit to enable MSI signaling. A device driver is prohibited from writing this bit to mask a device's service request. If 1, the device can use an MSI to request service. If 0, the device cannot use an MSI to request service.		0h	RW

35.11.1.29 Offset F4h: MADR – Message Signalled Interrupt Address Register

Table 35-147.Offset F4h: MADR: Message Signalled Interrupt Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** F4h **Offset End:** F7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ADDR	**Message Address:** Written by the system to indicate the lower 32-bits of the address to use for the MSI memory write transaction. The lower two bits will always be written as 0.		0h	RW

35.11.1.30 Offset F8h: MDATA – Message Signalled Interrupt Data Register

Table 35-148.Offset F8h: MDATA: Message Signalled Interrupt Data Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:7:0	**Offset Start:** F8h **Offset End:** F9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Message Data:** Written by the system to indicate the lower 16 bits of the data written in the MSI memory write DWORD transaction. The upper 16 bits of the transaction are written as 0.		0h	RW

35.12 Expansion Bus Configuration Space: Bus M, Device 8, Function 0

The Expansion Bus is Device 8 of Bus M, and is accessed using type 1 configuration cycles.

35.12.1 Register Details

Table 35-149.Bus M, Device 8, Function 0: Summary of Local Expansion Bus PCI Configuration Registers (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00h	01h	"Offset 00h: VID: Vendor Identification Register" on page 1320	8086h
02h	03h	"Offset 02h: DID: Device Identification Register" on page 1320	503Dh
04h	05h	"Offset 04h: PCICMD: Device Command Register" on page 1321	0000h
06h	07h	"Offset 06h: PCISTS: PCI Device Status Register" on page 1321	0010h
08h	08h	"Offset 08h: RID: Revision ID Register" on page 1322	Variable
09h	0Bh	"Offset 09h: CC: Class Code Register" on page 1323	068000h
0Eh	0Eh	"Offset 0Eh: HDR: Header Type Register" on page 1323	00h
10h	13h	"Offset 10h: CSRBAR: Control and Status Registers Base Address Register" on page 1323	00000000h
14h	17h	"Offset 14h: MMBAR: Expansion Bus Base Address Register" on page 1324	00000000h
2Ch	2Dh	"Offset 2Ch: SVID: Subsystem Vendor ID Register" on page 1325	0000h
2Eh	2Fh	"Offset 2Eh: SID: Subsystem ID Register" on page 1325	0000h
34h	34h	"Offset 34h: CP: Capabilities Pointer Register" on page 1326	DCh
3Ch	3Ch	"Offset 3Ch: IRQL: Interrupt Line Register" on page 1326	00h
3Dh	3Dh	"Offset 3Dh: IRQP: Interrupt Pin Register" on page 1326	01h
40h	43h	"Offset 40h: LEBCTL: LEB Control Register" on page 1327	00h
DCh	DCh	"Offset DCh: PCID: Power Management Capability ID Register" on page 1327	01h
DDh	DDh	"Offset DDh: PCP: Power Management Next Capability Pointer Register" on page 1328	E4h
DEh	DFh	"Offset DEh: PMCAP: Power Management Capability Register" on page 1328	0023h
E0h	E1h	"Offset E0h: PMCS: Power Management Control and Status Register" on page 1329	0000h
E4h	E4h	"Offset E4h: SCID: Signal Target Capability ID Register" on page 1329	09h
E5h	E5h	"Offset E5h: SCP: Signal Target Next Capability Pointer Register" on page 1330	F0h
E6h	E6h	"Offset E6h: SBC: Signal Target Byte Count Register" on page 1330	09h
E7h	E7h	"Offset E7h: STYP: Signal Target Capability Type Register" on page 1330	01h
E8h	E8h	"Offset E8h: SMIA: Signal Target IA Mask Register" on page 1331	00h
ECh	ECh	"Offset ECh: SINT: Signal Target Raw Interrupt Register" on page 1331	00h
F0h	F0h	"Offset F0h: MCID: Message Signalled Interrupt Capability ID Register" on page 1332	05h
F1h	F1h	"Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register" on page 1332	00h

Table 35-149.Bus M, Device 8, Function 0: Summary of Local Expansion Bus PCI Configuration Registers (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
F2h	F3h	"Offset F2h: MCTL: Message Signalled Interrupt Control Register" on page 1333	0000h
F4h	F7h	"Offset F4h: MADR: Message Signalled Interrupt Address Register" on page 1333	00000000h
F8h	F9h	"Offset F8h: MDATA: Message Signalled Interrupt Data Register" on page 1334	0000h

35.12.1.1 Offset 00h: VID – Vendor Identification Register

Table 35-150.Offset 00h: VID: Vendor Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 00h **Offset End:** 01h	
Size: 16 bit	**Default:** 8086h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	VID	**Vendor Identification:** This register field contains the PCI standard identification for Intel, 8086h.		8086h	RO

35.12.1.2 Offset 02h: DID – Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies any PCI device. Writes to this register have no effect.

Table 35-151.Offset 02h: DID: Device Identification Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 02h **Offset End:** 03h	
Size: 16 bit	**Default:** 503Dh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DID	**Device Identification Number:** This is a 16-bit value assigned to the Expansion Bus device.		503Dh	RO

35.12.1.3 Offset 04h: PCICMD – Device Command Register

Table 35-152.Offset 04h: PCICMD: Device Command Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 04h **Offset End:** 05h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	Reserved	Reserved		0h	RV
10	INTD	Interrupt Disable		0h	RW
09	FBTB	Fast Back-to-Back Enable		0h	RO
08	SER	SERR# Enable		0h	RO
07	Reserved	Reserved		0h	RV
06	PER	Parity Error Response		0h	RO
05	VPS	VGA Palette Snoop		0h	RO
04	MWE	Memory Write and Invalidate		0h	RO
03	SS	Special Cycle		0h	RO
02	BM	Bus Master Capable		0h	RW
01	MEM	Memory Space Enable: Setting this bit enables access to the memory regions the device claims through its BARs.		0h	RW
00	IO	**I/O Space Enable:** The device does not implement this functionality since it claims no I/O regions. The bit is hardwired to 0.		0h	RO

35.12.1.4 Offset 06h: PCISTS – Device Status Register

Table 35-153.Offset 06h: PCISTS: PCI Device Status Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	DPE	**Detected Parity Error:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
14	SSE	**Signaled System Error:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
13	RMA	**Received Master Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
12	RTA	**Received Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
11	STA	**Signaled Target Abort Status:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO

Table 35-153.Offset 06h: PCISTS: PCI Device Status Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** 06h **Offset End:** 07h	
Size: 16 bit	**Default:** 0010h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
10 : 09	DST	**DEVSEL Timing:** The device does not implement this functionality. These bits are hardwired to 0.		00b	RO
08	MDPE	**Master Data Parity Error Detected:** The device does not implement this functionality. The bit is hardwired to 0. The EP80579 uses signals for errors.		0h	RO
07	FB2B	**Fast Back-to-Back Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
06	Reserved	Reserved		0h	RV
05	MC66	**66 MHz Capable:** The device does not implement this functionality. The bit is hardwired to 0.		0h	RO
04	CL	**Capabilities List:** This bit is hardwired to 1 to indicate that the device has a capabilities list.		1	RO
03	IS	**Interrupt Status:**		0h	RO
02 : 00	Reserved	Reserved		0h	RV

35.12.1.5 Offset 08h: RID – Revision ID Register

The value of this register comes from the ICH Compatibility Rev ID registers.

Table 35-154.Offset 08h: RID: Revision ID Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** 08h **Offset End:** 08h	
Size: 8 bit	**Default:** Variable			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	RID	**Revision Identification Number:** This value indicates the revision identification number for the AIOC Device. The 4 most significant bits are always 0. The 4 least significant bits follow the ICH revision ID scheme as defined in Section 19.2.1.4, "Offset 08h: RID - Revision ID Register" on page 736.		Variable	RO

35.12.1.6 Offset 09h: CC – Class Code Register

Table 35-155.Offset 09h: CC: Class Code Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 09h **Offset End:** 0Bh	
Size: 24 bit	**Default:** 068000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 : 00	CC	**Class Code:** This value indicates the base class, subclass, and interface. 020000h = Network Controller / Ethernet controller		068000h	RO

35.12.1.7 Offset 0Eh: HDR – Header Type Register

Table 35-156.Offset 0Eh: HDR: Header Type Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 0Eh **Offset End:** 0Eh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	HDR	**PCI Header Type:** The header type of the device. 00h = single-function device with standard header layout.		0h	RO

35.12.1.8 Offset 10h: CSRBAR – Control and Status Registers Base Address Register

The CSRBAR is a PCI BAR in memory space that allows access to the control and status registers for the Local Expansion bus. See Chapter 42.0, “Register Summary” for a description of the registers this region exposes.

Table 35-157.Offset 10h: CSRBAR: Control and Status Registers Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 10h **Offset End:** 13h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region.		0h	RW
11 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 4KB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

35.12.1.9 Offset 14h: MMBAR – Expansion Bus Base Address Register

The MMBAR is a PCI BAR in memory space that allows access to the devices on the external Local Expansion bus. Expansion device 0 occupies the first 16MB/32MB of the region, device 1 occupies the next 16MB/32MB, etc. Refer to Section 42.0, "Local Expansion Bus Controller" for a description of the local expansion bus.

Table 35-158.Offset 14h: MMBAR: Expansion Bus Base Address Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 14h **Offset End:** 17h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 25	ADDR	**Upper Programmable Base Address:** These bits are set by BIOS to locate the base address of the region. Their behavior is described in Table 35-159.		0h	RW
24 : 04	ZERO	**Lower Bits:** Hardwired to 0 to set the region size to 32MB.		0h	RO
03	PREF	**Prefetchable:** Hardwired to 0 to indicate that the region is not prefetchable.		0h	RO
02 : 01	TYP	**Addressing Type:** Hardwired to 0 to indicate a 32-bit region.		00b	RO
00	MEM	**Memory Space Indicator:** Hardwired to 0 to identify the region as in memory space.		0h	RO

Table 35-159 defines the behavior of the ADDR field in MMBAR along with the resulting number of devices (chip selects) that the LEB supports for the configuration. The device count depends on the 16/32MB mode that the EXP_TIMING_CS[0-7] registers in the LEB controller select (refer to Section 42.5.1.1, "EXP_TIMING_CS0 - Expansion Bus Timing Register" and Section 42.5.1.2, "EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers"). Specifically, the LEB is in 32MB mode if bit 9 (this bit is in the CNFG_4_0 field) in *any* of these registers is set, otherwise it is in 16MB mode.

Table 35-159 defines the behavior of the ADDR field in MMBAR.

Table 35-159.MMBAR ADDR Field Behavior

LEBSIZE	ADDR Field		BAR Size	Number of LEB Devices Supported[a]	
	RW Bits	RO Zero Bits		16MB Mode	32MB Mode
0	None	31:25	Zero	None	None
1	31:25	None	32 MB	2	1
2	31:26	25	64 MB	4	2
3	31:27	26:25	128 MB	8	4
4	31:28	27:25	256 MB	8	8

a. The number of LEB devices supported depends on the device mode (16MB or 32MB) mode selected of the EXP_TIMING_CS[0-7] registers in the LEB controller.

35.12.1.10 Offset 2Ch: SVID – Subsystem Vendor ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-160.Offset 2Ch: SVID: Subsystem Vendor ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 2Ch **Offset End:** 2Dh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SVID	**Subsystem Vendor ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.12.1.11 Offset 2Eh: SID – Subsystem ID Register

This register is a write-once register. Once any byte in the register has been written, the register locks against further writes until reset.

Table 35-161.Offset 2Eh: SID: Subsystem ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 2Eh **Offset End:** 2Fh	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	SID	**Subsystem ID:** This field must be programmed during BIOS initialization.		0h	RWO

35.12.1.12 Offset 34h: CP – Capabilities Pointer Register

Table 35-162.Offset 34h: CP: Capabilities Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 34h **Offset End:** 34h	
Size: 8 bit	**Default:** DCh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	CP	**Pointer to First Capability Structure:** Value is DCh which is the config space offset of the first capability structure.		DCh	RO

35.12.1.13 Offset 3Ch: IRQL – Interrupt Line Register

Table 35-163.Offset 3Ch: IRQL: Interrupt Line Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 3Ch **Offset End:** 3Ch	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQL	**Interrupt Line:** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this device is connected to.		0h	RW

35.12.1.14 Offset 3Dh: IRQP – Interrupt Pin Register

Table 35-164.Offset 3Dh: IRQP: Interrupt Pin Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 3Dh **Offset End:** 3Dh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	IRQP	**Interrupt Pin:** Set to 01h to indicate the device always uses INTA# as its interrupt pin.		01h	RO

35.12.1.15 Offset 40h: LEBCTL – LEB Control Register

Table 35-165.Offset 40h: LEBCTL: LEB Control Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** 40h **Offset End:** 43h	
Size: 32 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 01	RSV	Reserved		0	RO
00	BSEN	**Byte Swap Control for LEB Controller.** '0' -> Byte swap not enabled '1' -> Byte swap enabled.		0	RW

35.12.1.16 Offset DCh: PCID – Power Management Capability ID Register

The Power Management Capability record controls power management in the device. It is a 6B PCI SIG-defined capability record and includes the PCID, PCP, PMCAP, and PMCS fields of the configuration header.

For an overview of the power management capability of AIOC devices, see Section 35.5, "Power Management of AIOC Devices".

Table 35-166.Offset DCh: PCID: Power Management Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** DCh **Offset End:** DCh	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCID	**Capability ID:** PCI SIG assigned capability record ID (01h, power management capability)		01h	RO

35.12.1.17 Offset DDh: PCP – Power Management Next Capability Pointer Register

Table 35-167.Offset DDh: PCP: Power Management Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** DDh **Offset End:** DDh	
Size: 8 bit	**Default:** E4h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	PCP	**Next Capability Pointer:** Hardwired to E4h to indicate the offset of the next capability.		E4h	RO

35.12.1.18 Offset DEh: PMCAP – Power Management Capability Register

Table 35-168.Offset DEh: PMCAP: Power Management Capability Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** DEh **Offset End:** DFh	
Size: 16 bit	**Default:** 0023h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 11	PME_SPT	PME# Support		0h	RO
10	D2_SPT	D2 Support		0h	RO
09	D1_SPT	D1 Support		0h	RO
08 : 06	AUX_CRNT	Aux Current		0h	RO
05	DSI	Device Specific Initialization		1	RO
04	RV	Reserved		0h	RV
03	PME_CLI	PME Clock		0h	RO
02 : 00	VER	Version		011b	RO

35.12.1.19 Offset E0h: PMCS – Power Management Control and Status Register

Table 35-169.Offset E0h: PMCS: Power Management Control and Status Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** E0h **Offset End:** E1h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	PME_STATUS	PME Status (sticky)		0h	RO
14 : 13	DATA_SCALE	Data Scale		00b	RO
12 : 09	DATA_SEL	Data Select		0000b	RO
08	PME_EN	PME Enable (sticky)		0h	RO
07 : 04	RV	Reserved		0000b	RO
03	NSR	No Soft Reset		0h	RO
02	RV	Reserved		0h	RO
01 : 00	PS	Power State		00b	RW

35.12.1.20 Offset E4h: SCID – Signal Target Capability ID Register

The Signal Target Capability record defines how the device targets it signals to IA agents. It is an 9B vendor-specific capability record and includes the SCID, SCP, SBC, STYP, SMIA, and SINT fields of the configuration header.

For more information on signaling by AIOC devices, see Section 35.4, “Interrupt Handling for AIOC Devices” on page 1235.

Table 35-170.Offset E4h: SCID: Signal Target Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** E4h **Offset End:** E4h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCID	**Capability ID:** PCI SIG assigned capability record ID (09h, vendor specific)		09h	RO

35.12.1.21 Offset E5h: SCP – Signal Target Next Capability Pointer Register

Table 35-171.Offset E5h: SCP: Signal Target Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** E5h **Offset End:** E5h	
Size: 8 bit	**Default:** F0h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SCP	**Next Capability Pointer:** Hardwired to F0h to indicate the offset of the next capability.		F0h	RO

35.12.1.22 Offset E6h: SBC – Signal Target Byte Count Register

Table 35-172.Offset E6h: SBC: Signal Target Byte Count Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** E6h **Offset End:** E6h	
Size: 8 bit	**Default:** 09h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	SBC	**Capability Record Byte Count:** Hardwired to the number of bytes in the vendor-specific capability record.		09h	RO

35.12.1.23 Offset E7h: STYP – Signal Target Capability Type Register

Table 35-173.Offset E7h: STYP: Signal Target Capability Type Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** E7h **Offset End:** E7h	
Size: 8 bit	**Default:** 01h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	STYP	**Capability Record Type:** Vendor assigned capability record type (01h, EP80579 signal target capability)		01h	RO

35.12.1.24 Offset E8h: SMIA – Signal Target IA Mask Register

Table 35-174.Offset E8h: SMIA: Signal Target IA Mask Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** E8h **Offset End:** E8h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved			0h	RW
01	SMIA1	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect Expansion Bus Parity Error		0h	RW
00	SMIA0	IA mask bit: If set to 1h, an interrupt is sent to the IA as either an INTx or MSI based on the PCI signaling configuration when detect Expansion Bus System Interrupt		0h	RW

35.12.1.25 Offset E9h: Reserved Register

Writing to this register will result in undefined behavior.

35.12.1.26 Offset EAh: Reserved Register

Writing to this register will result in undefined behavior.

35.12.1.27 Offset ECh: SINT – Signal Target Raw Interrupt Register

Table 35-175.Offset ECh: SINT: Signal Target Raw Interrupt Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** ECh **Offset End:** ECh	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 02	Reserved			0h	RO
01	SINT1	Interrupt: Read-only view of Expansion Bus Parity Error		0h	RO
00	SINT0	Interrupt: Read-only view of Expansion Bus System Interrupt		0h	RO

35.12.1.28 Offset F0h: MCID – Message Signalled Interrupt Capability ID Register

The Message Signalled Interrupt Capability record defines how the device generates PCI MSI messages. It is an 10B PCI SIG-defined capability record and includes the MCID, MCP, MCTL, MADR, and MDATA fields of the configuration header.

Table 35-176.Offset F0h: MCID: Message Signalled Interrupt Capability ID Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** F0h **Offset End:** F0h	
Size: 8 bit	**Default:** 05h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCID	**Capability ID:** PCI SIG assigned capability record ID (05h, MSI capability)		05h	RO

35.12.1.29 Offset F1h: MCP – Message Signalled Interrupt Next Capability Pointer Register

Table 35-177.Offset F1h: MCP: Message Signalled Interrupt Next Capability Pointer Register

Description:					
View: PCI	**BAR:** Configuration		**Bus:Device:Function:** M:8:0	**Offset Start:** F1h **Offset End:** F1h	
Size: 8 bit	**Default:** 00h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07 : 00	MCP	**Next Capability Pointer:** Hardwired to 0 to indicate this is the last capability.		0h	RO

35.12.1.30 Offset F2h: MCTL – Message Signalled Interrupt Control Register

Table 35-178.Offset F2h: MCTL: Message Signalled Interrupt Control Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** F2h **Offset End:** F3h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 09	Reserved	Reserved		0h	RO
08	MC	**Per-Vector Masking Capable:** Hardwired to 0 to indicate the device is not capable of per-vector masking.		0h	RO
07	C64	**64 bit Address Capable:** Hardwired to 0 to indicate the device does not generate 64b message addresses.		0h	RO
06 : 04	MME	**Multiple Message Enable:** System software writes to this field to indicate the number of allocated messages (less than or equal to the number of requested messages in MMC). A value of 0 corresponds to one message.		000h	RW
03 : 01	MMC	**Multiple Message Capable:** System software reads this field to determine the number of requested messages. Hardwired to 0 to request one message.		000h	RO
00	MSIE	MSI Enable: System software sets this bit to enable MSI signaling. A device driver is prohibited from writing this bit to mask a device's service request. If 1, the device can use an MSI to request service. If 0, the device cannot use an MSI to request service.		0h	RW

35.12.1.31 Offset F4h: MADR – Message Signalled Interrupt Address Register

Table 35-179.Offset F4h: MADR: Message Signalled Interrupt Address Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** F4h **Offset End:** F7h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ADDR	**Message Address:** Written by the system to indicate the lower 32-bits of the address to use for the MSI memory write transaction. The lower two bits will always be written as 0.		0h	RW

35.12.1.32 Offset F8h: MDATA – Message Signalled Interrupt Data Register

Table 35-180.Offset F8h: MDATA: Message Signalled Interrupt Data Register

Description:					
View: PCI	**BAR:** Configuration	**Bus:Device:Function:** M:8:0		**Offset Start:** F8h **Offset End:** F9h	
Size: 16 bit	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 : 00	DATA	**Message Data:** Written by the system to indicate the lower 16 bits of the data written in the MSI memory write DWORD transaction. The upper 16 bits of the transaction are written as 0.		0h	RW

§ §

36.0 AIOC Interfaces

36.1 Overview

The AIOC interfaces include:

- Gigabit Ethernet MACs
- TDM interface
  Note: This interface is only available on the Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology. See Table 47-1, “Base Features of EP80579 SKUs” on page 1730.
- Local Expansion Bus interface
- Controller Area Network interfaces
- IEEE-1588 time synchronization hardware assist unit
- Synchronous Serial Port (SSP)

The details for the TDM interface are available in the software for the Intel® EP80579 Integrated Processor product line enabling documentation.

36.2 Gigabit Ethernet (GbE)

The three Gigabit Ethernet controllers off-load various tasks from the IA-32 core to improve performance. This off-load is often times referred to as “Network Interface Card” or NIC functionality.

Provided features fall into these five general categories:

- MAC features
- Host off-loading
- Link interface
- Power Management (Wake On LAN)
- Serial EEPROM Interface
- Integrated DMA

The GbE controller is based on an Intel fourth-generation Gigabit MAC to provide a standard IEEE 802.3 Ethernet interface for 1000BASE-T, 100BASE-T, and 10BASE-T applications (802.3, 802.3u, and 802.3ab). The controller is capable of transmitting and receiving data rates of 1,000 Mbps, 100 Mbps or 10 Mbps.

By using hardware acceleration, the controller is able to off-load tasks, such as checksum calculations and TCP segmentation from the host processor.

The native interfaces supported by the GbE controller are MII and GMII. Protocol translation gaskets are used to provide the RMII and RGMII interfaces.

Power management features include support for APM and APCI. Each controller supports the D0 and D3 states including internal clock gating for reduced power consumption in the D3 state. Further, one controller is located in an independent auxiliary power well for particularly power sensitive applications.

The DMA function provides the capability for packet data transfer and descriptor management. The GbE controller is able to cache up to 64-packet descriptors. A 64-KByte, on-chip packet buffer maintains performance as available bandwidth changes.

Finally, each controller is capable of self-configuration via an optional, external serial EEPROM.

36.2.1 Integrated DMA Features

- Performs descriptor-driven receive packet transfers from Rx MAC to system memory.
- Performs descriptor-driven transmit packet transfers from system memory to Tx MAC.
- Separate transmit and receive DMA engines.
- 64-entry descriptor caches for transmit and receive.
- Descriptor ring management hardware.
- Configurable 64KB packet buffer.

36.2.2 MAC Features

The MAC controller's SCSMA/CD unit handles all the IEEE 802.3 receive and transmit MAC functions while interfacing between the DMA and link interface (RMII/RGMII) The MAC unit supports:

- Complete CSMA/CD function supporting IEEE 802.3 (10Mb/s), 802.3u (100Mb/s), 802.3z and 802.3ab (1000Mb/s).
- Half- and full-duplex operation at 10/100.
- Full-duplex operation at 1000 Mbps.
- Up to 16 addresses for exact match unicast/multicast address filtering.
- Multicast address filtering based on 4,096-bit vectors in addition to promiscuous unicast and promiscuous multicast filtering.
- MAC strips IEEE 802.1q VLAN tags and filters packets based on their VLAN ID. Up to 4,096 VLAN tags are supported.
- Transmit path supports insertion of VLAN tag information on a packet-by-packet basis.
- Flow control as defined in IEEE 802.3x as well as specific operation of asymmetrical flow control defined by IEEE 802.3z and software controllable pause times and threshold values.
- Programmable host memory receive buffers (256 Byte to 16 KByte) and cache line size (16 to 256 Byte).
- 16 KByte configurable transmit and receive FIFO buffers with ECC protection.
- Wakeup.
- ACPI down functionality supporting D0 & D3 states.

36.2.3 Host Off-Loading Features

The GbE controller provides the ability to off-load IP, TCP, and UDP checksum for transmit, packet filtering, and hardware VLAN support. The functionality provided by these features significantly reduce the IA processor utilization by shifting the burden of the functions from the driver to the hardware.

- IPV4 checksum calculation
- TCP/UDP checksum calculation
- 16 KByte jumbo-frame support
- Interrupt coalescing (multiple packets per interrupt)
- TCP segmentation for transmits that off-loads packet segmentation and encapsulation
- RMON statistics
- IPV6 support for IP/TCP and IP/UDP receive checksum off-load
- IP v6 wake-up filters
- IP v6 TCP segmentation

36.2.4 Interfaces

The GbE controller provides the following serial interfaces:

- RMII — 8-pin interface
- RGMII — 12-pin interface
- MDIO/MDC Interface

Note: For information on PHY selection:

1. Go to http://www.intel.com/go/soc.
2. Select the "Technical Documentation" tab.
3. Search for "Ethernet PHY Selection Criteria for the Intel® EP80579 Integrated Processor product line".

36.2.5 Power Management

Each controller supports the following Power Management related features:

- Power states of D0 & D3hot with optional D3cold support.
  - — D3cold support on GbE0 only.
- Wake On LAN.

Each GbE may be configured to generate a PME signal on the reception of a Magic packet or a network wake-up packet. A combination of pre-defined and user configurable filters are provided to support this functionality. The PME_N wake-up signals from all GbE units are wired-OR together and brought out to a single external pin. This pin should be externally wired to the PME_N pin of the ICH.

When transitioned to the D3hot state, the GbE controller internally performs clock-gating on many of its blocks to reduce power.

Additionally, one of the GbE controllers (GbE0) is located in the auxiliary power well. This allows monitoring of LAN traffic on this interface while the rest of the EP80579 is in ultra-low-power mode. The details of power well manipulation is outside the scope of this chapter and the reader is referred to Chapter 37.0, “Gigabit Ethernet Controller” for details.

36.2.6 Serial EEPROM Interface

A single four-wire Microwire* interface is provided for connection of an optional, externally connected serial EEPROM. The serial EEPROM may be used to provide configuration information to the GbE controllers upon power-up or reset. All three controllers share the same EEPROM. A fixed priority arbiter controls access to the EEPROM where highest priority is given to GbE 0 then GbE 1 and finally GbE 2.

Software may also access this EEPROM. It can either use the controller built in to the MAC to read the EEPROM, or access the EEPROM directly using the EEPROM's 4-wire interface via registers also provided in the MAC.

36.3 Local Expansion Bus Interface (LEB)

The Local Expansion Bus Controller (LEB) provides a low-speed interface to external expansion target devices.

The Expansion Bus Controller includes a 25-bit address bus and a 16-bit-wide data path. The Expansion Bus controller maps transfers between the EP80579 and external devices. The Expansion Bus supports these target devices:

- Intel multiplexed
- Intel non-multiplexed
- Intel StrataFlash® technology
- Intel StrataFlash® Synchronous Memory
- Micron* Flow-Through ZBT
- Motorola* multiplexed
- Motorola* non-multiplexed
- Texas Instruments* Host Port Interface* (HPI)

Applications having less than 16-bit, external target devices may connect to an 8-bit interface. For TI DSPs that support an internal bus width of 32 bits, the multiplexed HPI-8 or HPI-16 interface can be used to complete these transfers.

The Expansion Bus Controller features include:

- Outbound transfers — the EP80579 is the master to an external target device.
- Eight programmable target chip selects.
- Twenty-five bits of address; 16 bits of data.
- Supports Intel-mode and Motorola-mode bus cycles.
- Supports Intel StrataFlash.
- Supports 66-MHz Synchronous Intel StrataFlash Memory (16-bit only).
- Supports 16-bit Micron* Flow-Through ZBT (Zero bus turnaround) SRAMS.
- Supports 8-bit and 16-bit Texas Instruments* *HPI specifications*.
- Multiplexed or non-multiplexed address/data buses for Intel/Motorola/HPI bus cycles.

- Supports even- and odd-parity generation and calculation.
- Maximum clock input frequency of 80 MHz.

36.4 Serial Synchronous Port (SSP)

The SSP is a full-duplex, synchronous, serial interface. It can connect to a variety of external analog-to-digital (A/D) converters, audio and telecom CODECs, and many other devices that use serial protocols for transferring data.

The interface supports National Microwire*, Texas Instruments* Synchronous Serial Protocol (SSP), and Motorola* Serial Peripheral Interface (SPI) protocol.

Key features of the SSP Port are:

- Operates in master mode (the attached peripheral functions as a slave).
- Supports serial bit rates from 7.2 Kb/s to 1.84 Mb/s.
- Serial data formats may range from 4 to 16 bits in length.
- Two on-chip register blocks function as independent FIFOs for data, one for each direction.
- The buffers are 16 entries deep x 16 bits wide.

Buffers may be burst-loaded or emptied by the system processor using SRAM-like burst transfers, from 1 to 8 words per transfer. Each 32-bit word from the system fills one entry in a FIFO using the lower half 16-bits of a 32-bit word.

36.5 Controller Area Network (CAN)

The Controller Area Network (CAN) is a serial bus system used in a broad range of embedded as well as automation control systems. It usually links two or more microcontroller-based physical devices.

CAN protocol is based on a broadcast communication mechanism. This broadcast communication is achieved by using a message-oriented transmission protocol. In this protocol, station and station addresses are not defined. Only messages are defined.

These messages are identified by using a message identifier. The message identifier has to be unique within the whole network and it defines not only the content but also the priority of the message.

A high degree of system and configuration flexibility is achieved as a result of the content-oriented addressing scheme.This content-oriented addressing scheme allows for a high degree of system and configuration flexibility.

It is very easy to add stations to a existing CAN network without making any hardware or software modifications to the existing stations as long as the new stations are purely receivers. This allows the concept of modular electronics and also permits multiple reception and the synchronization of distributed processes. Data needed as information by several stations can be transmitted via the network in such a way that it is unnecessary for each station to have to know who the producer of the data is. This allows easy servicing and upgrading of networks as data transmission is not based on the availability of specific types of stations.

Key features include:

- Support for CAN 2.0B protocol
- Support for 11-bit and 29-bit identifiers
- Bit rates up to 1 Mbps

- Clock frequency of 40 MHz

36.6 IEEE 1588 Time Synchronization Hardware Assist

In a distributed control system containing multiple clocks, individual clocks tend to drift apart. Some kind of correction mechanism is necessary to synchronize the individual clocks to maintain global time, which is accurate to some requisite clock resolution. The IEEE 1588 standard for a precision-clock-synchronization protocol for networked measurement and control systems can be used for this purpose.

The IEEE standard defines several messages that can be used to exchange timing information. The hardware-assist logic required to achieve precision clock synchronization using the IEEE 1588 standard is left to implementation.

The time-synchronization logic monitors the internal MII/GMII signals on two of the three Gigabit Ethernet interfaces. Additionally, this block monitors both CAN interfaces.

An interrupt signal to the IA-32 core is generated when any of the following conditions occur and are enabled:

- Target time expiration
- Auxiliary Master Mode snapshot is taken
- Auxiliary Slave Mode snapshot is taken

The System Time and Target Time registers are both 64 bits wide. When the system time is greater than or equal to the target time, the target time expiration condition will set. An interrupt enable mask must be set to allow the target time interrupt to pass to the core.

§ §

37.0 Gigabit Ethernet Controller

37.1 Overview

The three Gigabit Ethernet controllersoff load various tasks from the IA-32 core to improve performance. This off load is often times referred to “Network Interface Card” or NIC functions. The Gigabit Ethernet controller is based on an Intel fourth generation Gigabit MAC to provide a standard IEEE 802.3 Ethernet interface for 1000BASE-T, 100BASE-TX, and 10BASE-T applications. The controller is capable of transmitting and receiving data rates of 10/100/1000 Mbps. The Convergence and PMD Sublayer communication of the Physical Layer are implemented in MII/GMII (internally) and RMII/RGMII (externally).

Note: The RGMII, and RMII, while not natively supported in the Gigabit Ethernet controller are supported by protocol translators between the Gigabit Ethernet controller and the pins.

Note: For information on PHY selection:

1. Go to http://www.intel.com/go/soc.
2. Select the "Technical Documentation" tab.
3. Search for "Ethernet PHY Selection Criteria for the Intel® EP80579 Integrated Processor product line”.

37.1.1 Terminology and Conventions

37.1.1.1 Register and Bit References

This document refers to device register names with all capital letters. To refer to a specific bit in a register the convention REGISTER.BIT is used. For example, CTRL.BEM refers to the Big Endian Mode bit in the Device Control Register (CTRL).

37.1.1.2 Byte and Bit Designations

This document uses “B” to abbreviate quantities of bytes, i.e. a 4KB represents 4096 bytes. Similarly, “b” is used to represent quantities of bits, i.e. 100Mbps represents 100 Megabits per second.

37.1.1.3 Numbering

All numbers are in decimal unless otherwise indicated. Hexadecimal numbers will be preceded by a 0x or followed by an h. If not clear from the context, binary numbers will be followed by a b or (b). Single bits are generally given as simply 0 or 1, since the value is identical in binary or decimal.

37.1.1.4 Memory Alignment Terminology

Some GbE data structures have special memory alignment requirements. This implies that the starting physical address of a data structure must be aligned as specified in this GbE chapter. The following terms are used for this purpose:

- BYTE alignment implies that the physical addresses can be odd or even. Examples: 0FECBD9A1h, 02345ADC6h.
- WORD alignment implies that physical addresses must be aligned on even boundaries; i.e., the last nibble of the address may only end in 0, 2, 4, 6, 8, Ah, Ch, or Eh. For example, 0FECBD9A2h.
- DWORD (Double-Word) alignment implies that the physical addresses may only be aligned on 4-byte boundaries; i.e., the last nibble of the address may only end in 0, 4, 8, or Ch. For example, 0FECBD9A8h.
- QWORD (Quad-Word) alignment: Implies that the physical addresses may only be aligned on 8byte boundaries; i.e., the last nibble of the address may only end in 0, or 8. For example, 0FECBD9A8h.
- PARAGRAPH alignment implies that the physical addresses may only be aligned on 16-byte boundaries; i.e., the last nibble must be a 0. For example, 02345ADC0h.

37.1.1.5 Alignment and Byte Ordering

It should be noted that the data stream in Ethernet has no notion of byte alignment. All data on the wire is referenced as bit ordered. Data is presented “on the wire” least significant bit (lsb) first.

For example: A big-endian destination address in the Ethernet header may be represented in as 0x00_11_22_33_44_55. In this case, the data is seen on the wire in the byte order as written from left to right, however the bits are seen lsb first, i.e. bit 0 of the first “00” byte occurs first.

Representations of these fields internal to the EP80579 may reverse the byte ordering as shown above. Care must be taken to avoid byte ordering errors when programming the device.

Refer to Section 37.5.14, “Endianness” on page 1422 and Section 3.6, “Endianness” on page 119.

37.1.1.6 Packet Buffer

The GbE Packet Buffer (PB) is an ECC protected 64KB dedicated memory used for buffering transmit and receive packets as they are communicated. The proportions of the PB dedicated to TX and RX operations is software configurable.

Throughout this chapter, the TX portion of PB is referred to as the TX FIFO and the RX portion of the PB is referred to as the RX FIFO. These terms are interchangeable.

It is also worthwhile to note that any information in the PB, either for incoming or outgoing data, is volatile. The GbE will receive packets into the RX PB and transfer the RX descriptor(s) and payload into host memory for the host CPU before sending interrupt notification. Conversely, the host CPU will transfer TX descriptor(s) and payload into host memory for the GbE before notifying the GbE that the packet information is available. After this notification, the GbE transfers the TX information to the TX PB and sends the data out on the wire. The host CPU will never have to directly access the GbE’s PB, however diagnostic access is supported.

37.1.2 Wake On LAN

The GbE does not support preservation of Magic Packets for Wake On LAN. Additionally, the storage of other packets received after a qualifying wake event is not supported until the GbE has been returned to a D0a state. If the GbE was in D3 then it must be specifically moved to D0 by the IA software writing to the Power Capability register or Reset must be cycled and the GbE re-initialized. Software intervention is required to move the GbE to D0a and restart packet receive processing.

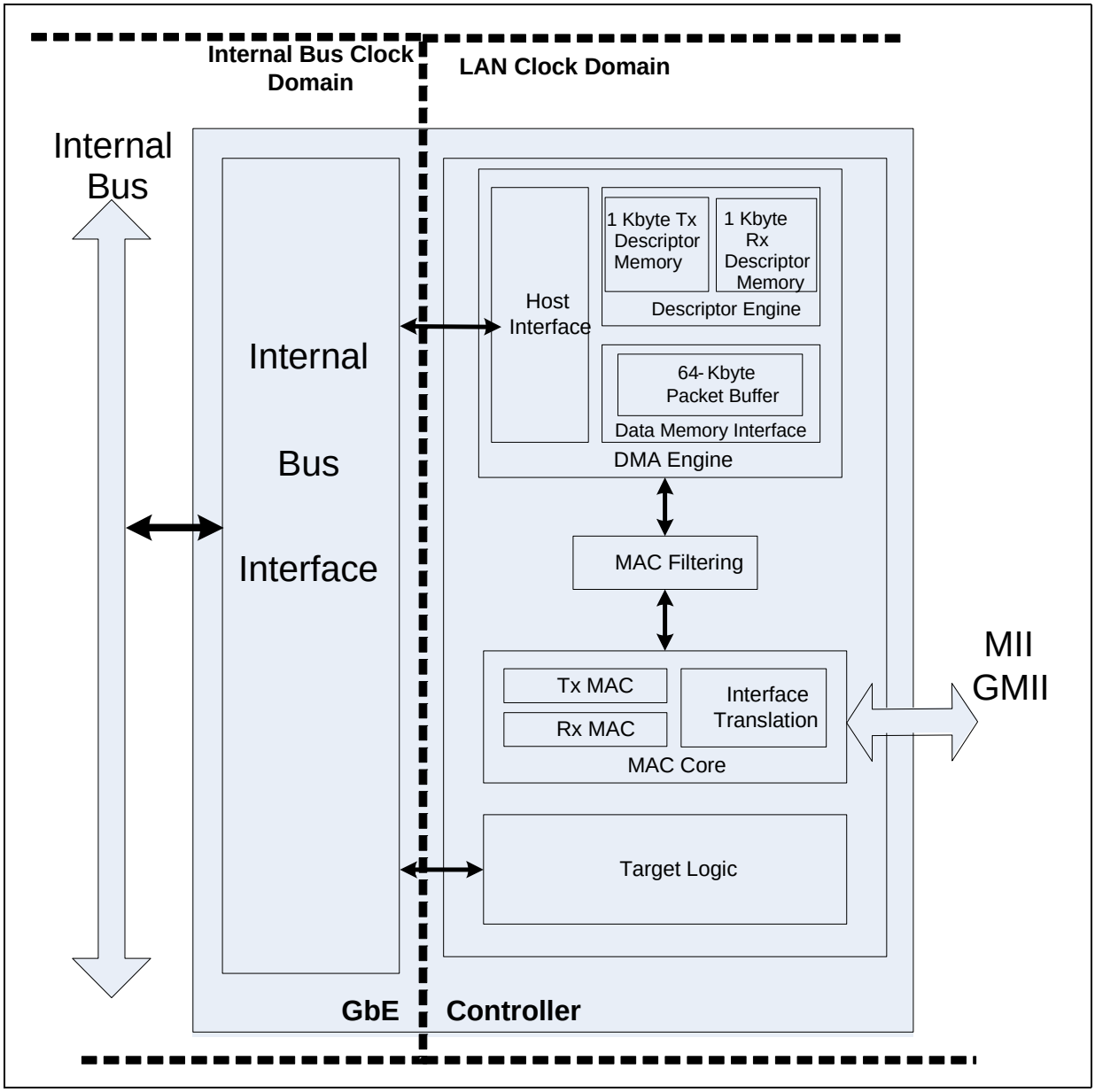
37.2 Feature List

The following is a list of GbE MAC features:

- Compliant with the 1000 Mbps Ethernet/802.3z specification.
- Half-duplex MII operation.
- Full-duplex operation at all MII and GMII speeds.
- Programmable system memory receive buffers (256B to 16KB).
- 64B cache-line size.
- 64KB RX/TX packet buffer (defaulted to 48 KB RX and 16 KB TX, but configurable).
- Descriptor ring management hardware for TX packet buffer.
- Support for little-endian or big-endian byte ordering.
- Flow Control Support: send/receive PAUSE frames & receive FIFO thresholds.
- Statistics for Local Management and Software Optimization.
- A mechanism for delaying/reducing transmit interrupts.
- Functional and Error interrupt signals.
- Software-controlled reset bit.
- Frame formats supported by the EP80579's GbE include:
  - Ethernet 802.3
  - IEEE 802.1q VLAN (Ethernet 802.3ac)
  - Ethernet Type 2
  - Ethernet SNAP
  - IPv4 headers with options
  - IPv6 headers with IP option next headers.
  - IPv6 packet tunneled in IPv4.
  - TCP with options.
  - UDP with options.
- Multi-speed operation: 10/100/1000 Mbps.
- All local memories and buffers are protected.

37.3 Functional Block Diagram

Figure 37-1. GbE Controller Block Diagram



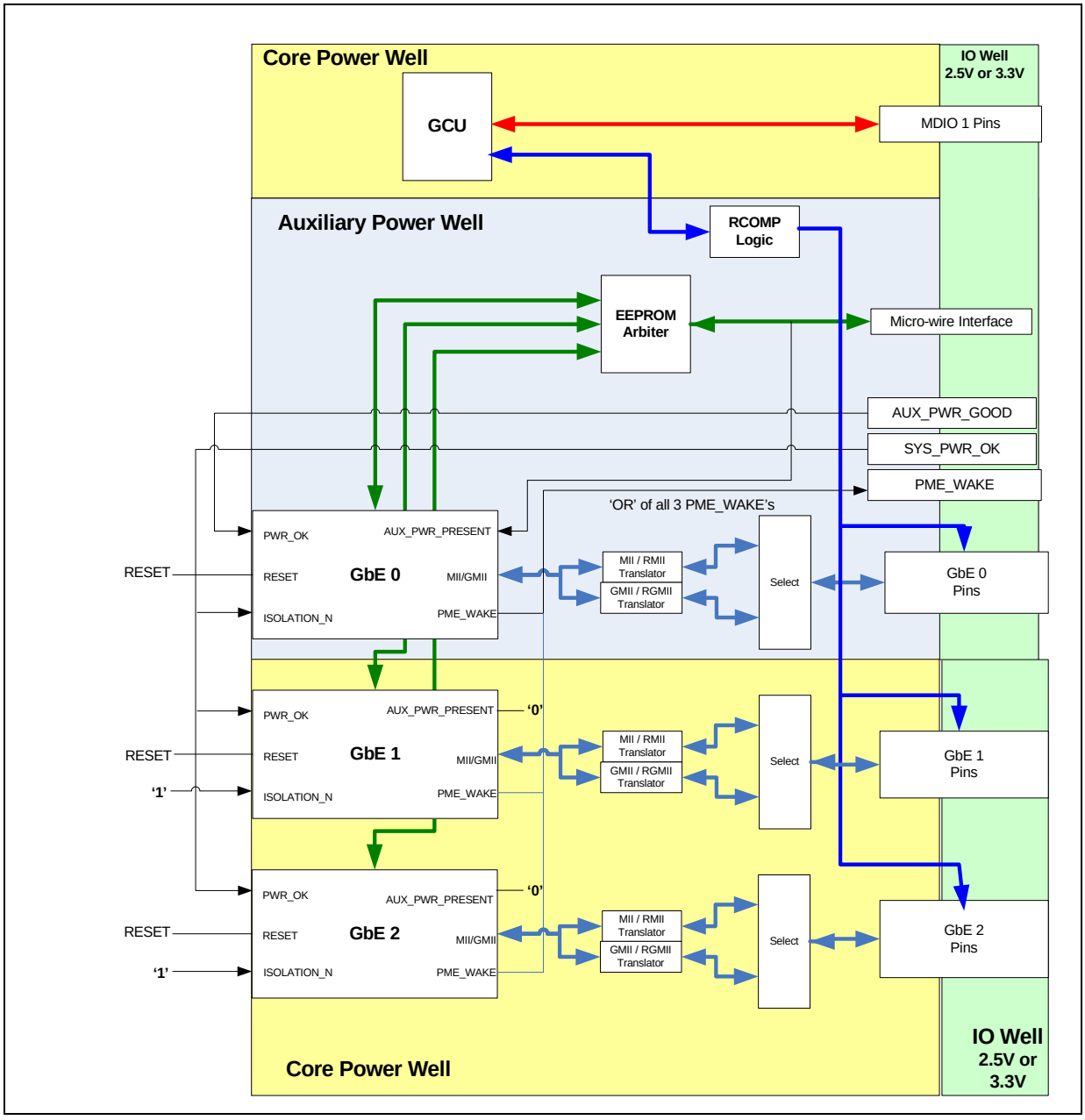
The internal bus interfaces provide the access to the GbE CSRs as well as the medium over which ethernet packets are transferred between the Ethernet client and the EP80579.

The DMA Engine buffers the packets between system memory and the MAC Core. The MAC Core transfers packets between the DMA Engine and the Ethernet Client. The internal bus interface accepts the PCI Configuration transactions and routes those transactions to the proper internal module. These blocks all operate in the LAN clock domain.

37.4 Usage Model

The EP80579 implements three GbE controllers. In addition to the three controllers, protocol translation gaskets and an MDIO interface are provided. Also, a multiplexor to allow the three GbE controllers to share access to a common serial EEPROM is implemented. A block diagram of these units is shown in Figure 37-2:

Figure 37-2. GbE Ethernet Complex



37.4.1 Protocol Translation

While the GbE's does not natively support RGMII or RMII a protocol translator is implemented between the GbE and the pins that converts RGMII to GMII and RMII to MII for the GbE. Even though the pins may be using an RGMII or RMII protocol the GbE is still only communicating in its native GMII and/or MII protocols.

The selection of the appropriate translator is controlled by the Auxiliary Device Control Register. Refer to Section 37.6.2.4, "CTRL_AUX – Auxiliary Device Control/Status Register"

Note that when the translator selection register has enabled RGMII, the GbE must be configured to GMII/MII operation. Likewise, when the settings have enabled RMII, the GbE must be configured to MII operation.

37.4.2 Power Management

As shown in Figure 37-2, "GbE Ethernet Complex" on page 1345, GbE0 is located in a separate auxiliary power well. This gives the user the ability to bring the rest of the chip into an ultra-low power mode (power removed) while GbE0 continues to operate in D3cold to support Wake On LAN. If a separate auxiliary power source is used for GbE0, then platform must also supply an Aux_Pwr_Good signal which is asserted to indicate that the auxiliary power supply is stable and that the reference clock for the GbE is stable. Additionally, the EEDI pin should be pulled-up to provide the internal aux_pwr_present signal indicating that an auxiliary supply is being used. If an auxiliary power supply is not being used and the power supplied to the auxiliary well is the same as that for the core logic, then EEDI must be tied low. In this situation the same signal as provided to Sys_Pwr_OK is also routed to Aux_Pwr_Good.

Note that if an auxiliary supply is used for GbE0 then the corresponding PHY and also the serial EEPROM, if present, must be powered.

Refer to the Section 6.3.2, "Power Management Support"and Section 37.5.10, "Wake on LAN" for more details.

37.4.3 Software Initialization and Diagnostics

This section discusses general software notes for the GbE, especially initialization steps. This includes general hardware power-up state, basic device configuration, initialization of transmit and receive operation, link configuration, software reset capability, statistics, and diagnostic hints.

37.4.3.1 Power Up State

When the MAC powers up it will read the optional EEPROM. The EEPROM contains sufficient information to bring the link up and configure the MAC for manageability and/or APM wakeup. However, software initialization is required for normal operation and if the EEPROM is not present.

The power-up sequence, as well as transitions between power states, is described in Section 37.7.3, "Power States" on page 1550. The detailed timing is given in Section 37.7.4, "Timing of Power-State Transitions" on page 1551. Section 37.6, "GbE Controller Register Summary" on page 1425 gives detailed registers descriptions with details on power on defaults.

37.4.3.2 Memory Initialization

The GbE's internal memory must be initialized after the GbE unit has been powered-up, to insure that no ECC or parity errors are generated from the random state of memory after a power-up. If an ECC or parity error is encountered, a soft reset must be issued, then the entire affected memory (that reported the error) must be re-initialized to remove the error, so that it doesn't re-occur.

After each power-up and soft reset, and after the memories have been initialized as described below, software must clear the appropriate ECC disable bits described in Section 37.6.8.3, "MEM_STS – Memory Error Status Register" on page 1546. Neither ECC nor parity errors will be reported nor will the GbE error handling be enabled (as described in Section 37.5.12, "Error Handling" on page 1418) until these bits have been cleared out. These may be cleared immediately after a soft reset if the error condition that resulted in the soft reset did not involve an ECC or parity error from one of these memories.

37.4.3.2.1 Packet Buffer Memory

The Packet Buffer memory is initialized through write accesses to the PBM locations. A single PBM access will initialize four locations within the Packet Buffer memory, using address bits 15:4 of the PBM access to select the 128bit Packet Buffer Memory location. Therefore, 4096 writes are required to initialize the 64KB of packet memory, writing every four PBM locations (PBM(0), PBM(4), PBM(8), etc.).

37.4.3.2.2 Descriptor Tx and Rx Memory

The Descriptor Tx and Rx memories are each initialized through write accesses to the TXDESCM and RXDESCM locations. A single DESCTX or DESCRX access will initialize four locations within the Descriptor memory, using address bits 15:4 of the access to select the 128bit Descriptor Memory location. Therefore, 64 writes are required to initialize the 1KB of descriptor memory, writing every four DESC locations (DESCRX(0), DESCRX(4), DESCRX(8), etc.).

37.4.3.2.3 Multicast Filter and Special Packets Memory

The Multicast and VLAN filter memories are each initialized through write accesses to the MTA and VFTA locations as described in Section 37.6.4.14, "MTA[0-127] – 128 Multicast Table Array Registers" on page 1488 and Section 37.6.4.17, "VFTA[0-127] – 128 VLAN Filter Table Array Registers" on page 1490. Initialization will require 128 writes to initialize the 512B of descriptor memory, writing every MTA or VLAN filter location.

37.4.3.2.4 Flexible Filter Memory

The Flexible Filter memories are each initialized through write accesses to the FFVT and FFMT locations as described in Section 37.6.7.12, "FFVT[0-127] – Flexible Filter Value Table Registers" on page 1543 and Section 37.6.7.11, "FFMT[0-127] – Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1542.

Initialization will require 128 writes to initialize the 512B of descriptor memory, writing every MTA or VLAN filter location. Initialization will require 256 writes to initialize the memories, writing every one of the 128 FFMT and FFVT locations.

37.4.3.2.5 Statistics Memory

The Statistics memory provides no write access, but is initialized by the GbE automatically after a power-up reset or software reset.

37.4.3.3 General Configuration

Several values in the CTRL – Device Control Register, CTRL_EXT – Extended Device Control Register and CTRL_AUX – Auxiliary Device Control/Status Register need to be set upon power up or after a device reset for normal operation.

- Convergence Layer Mode of the MAC is programmed by the CTRL_EXT.LINK_MODE setting. This value may also be read by software from the STATUS – Device Status Register at STATUS.TBIMODE.
- Duplex mode is determined via Auto-Negotiation between the external PHY and it's link partner. Software either continuously polls the PHY registers via MDIO until a link is detected or the host CPU is interrupted when the link is established via the PHY's MDINT capability. Software then programs CTRL.FD per the interface negotiation. Status information can be found at STATUS.FD.
- Speed is determined via Auto-Negotiation between the external PHY and it's link partner. Software either continuously polls the PHY registers via MDIO until a link is detected or the host CPU is interrupted when the link is established. Software then programs CTRL.SPEED per the interface negotiation. Status information can be found at STATUS.SPEED.
- Desired endianness configuration must be set in CTRL_AUX.

37.4.3.4 Link Setup Mechanisms and Control/Status Bit Summary

- MAC duplex and speed settings forced by software based on resolution of PHY
  - — CTRL.FD and CTRL.SPEED is set by software based on reading PHY status register after PHY has successfully auto-negotiated a link with the link partner.
  - — CTRL.RFCE and CTRL.TFCE must be set by software after reading flow control resolution from PHY registers

37.4.3.5 Receive Initialization

Software must program the Receive Address Low Register (RAL) and Receive Address High Register (RAH) registers to represent the receive address(es) per the station address. The station address (a.k.a the MAC address) is RAL/RAH(0), fifteen additional receive addresses can be programmed in addition to this.

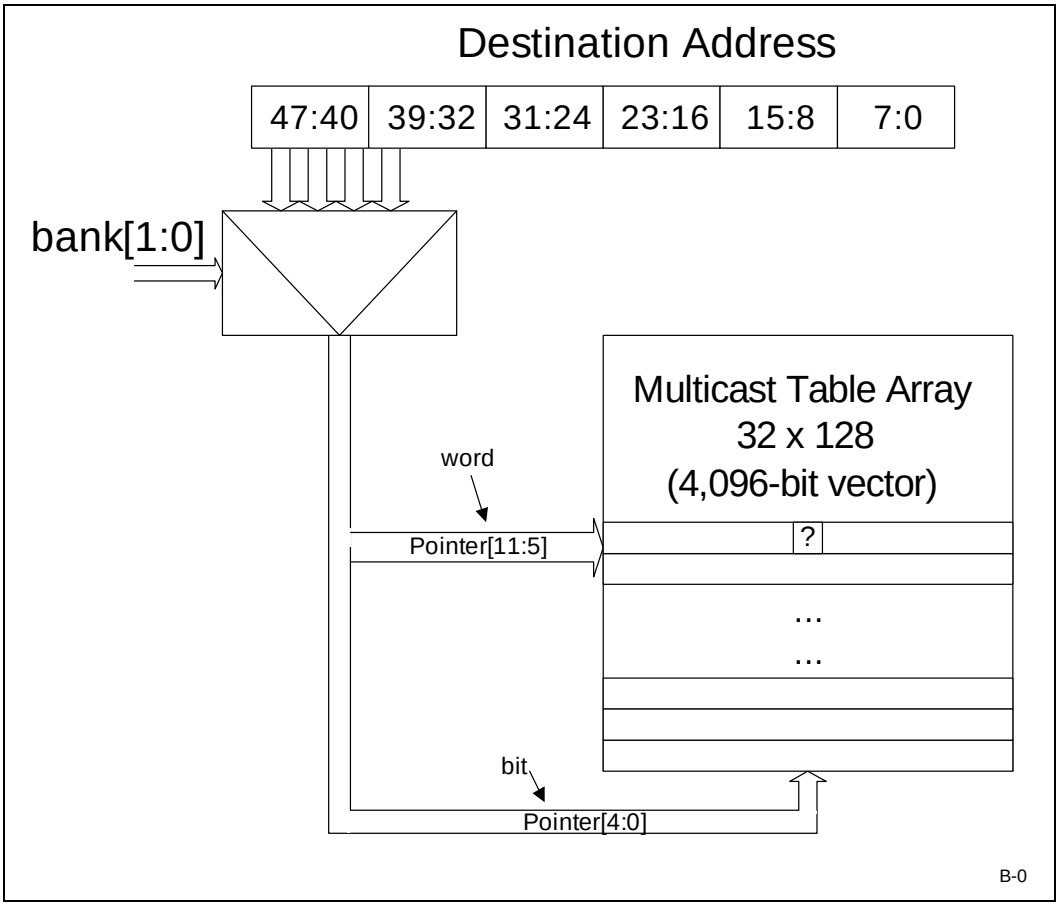
Software must also set up the 128 Multicast Table Array Registers (MTA[127:0]). This probably means zeroing all entries initially and adding in entries to the Multicast Table Array as requested.

Figure 37-3 diagrams the multicast lookup algorithm. The destination address shown represents the internally stored ordering of the received Destination Address. Note that bit 0 indicated in this diagram is the first on the wire. Refer to "Ethernet Addressing" on page 1351 for more details. Note that the bank bits is the RCTL.MO setting.

Program the Interrupt Mask Set/Read Register (IMS) to pass any interrupt the driver cares about to the Interrupt Controller for routing further to the EP80579's IA-32 core. Suggested bits include IMS.RXT0, IMS.RXO, and IMS.RXDMT0.

Program Receive Control Register (RCTL) with appropriate values. If initializing, it is necessary to leave the receive logic disabled, RCTL.EN = 0, until after the receive descriptor ring has been initialized. If VLANs are not used, software should clear RCTL.VFE. Then there is no need to initialize the 128 VLAN Filter Table Array Registers (VFTA[127:0]). Initialize the remainder of RCTL as desired, refer to "Section 37.6.4.1, "RCTL – Receive Control Register"" or details.

Figure 37-3. Multicast Table Array Algorithm



To properly receive packets requires only that the receiver is enabled. This should be done only after all other setup is accomplished. If software uses the Receive Descriptor Minimum Threshold Interrupt (IMS.RXDMT0 = 1), that Receive Threshold value should be set in the Flow Control Receive Threshold Low Register (FCRTL) and Flow Control Receive Threshold High Register (FCRTH) MMRs.

Allocate a contiguous region of memory for the receive descriptors. Program the receive descriptor region into the following MMRs describing the memory region:

- Receive Descriptor Base Address Low Register (RDBAL)
- Receive Descriptor Base Address High Register (RDBAH)
- Receive Descriptor Length Register (RDLEN)

The Receive Descriptor Head Register (RDH) and Receive Descriptor Tail Register (RDT) pointers are initialized (by hardware) to 0 after a power-on or a software-initiated device reset. Receive buffers of appropriate size should be allocated using RCTL.BSEX and RCTL.BSIZE. Pointers to these buffers should be stored in the descriptor ring. The tail pointer should be set to point one descriptor beyond the end.

37.4.3.6 Transmit Initialization

Packet transmission is configured via the Transmit Control Register (TCTL). For example, values for this MMR could be:

- TCTL.CT = 0x0F
- TCTL.COLD = 0x200 (512d); for half-duplex mode
- TCTL.COLD = 0x040 (64d); for full-duplex mode
- TCTL.PSP = 1

Note: Not all of these values are needed in all duplex modes, but it is more concise to simply always program them to the values shown regardless of mode.

Program the Transmit IPG Register (TIPG) with the following values to get the minimum legal inter packet gap (IPG):

- TIPG.IPGT = 0x8 (8d)
- TIPG.IPGR1 = 0x8 (8d)
- TIPG.IPGR2 = 0x6 (6d)

Note: Not all of these values are needed in all duplex modes, but it is more concise to simply always program them to the values shown regardless of mode.

Allocate a contiguous region of memory for the transmit descriptor list. Program the transmit descriptor region into the following MMRs describing the memory region:

- Transmit Descriptor Base Address Low Register (TDBAL),
- Transmit Descriptor Base Address High Register (TDBAH),
- Transmit Descriptor Length Register (TDLEN),
- Transmit Descriptor Head Register (TDH), and
- Transmit Descriptor Tail Register (TDT).

37.4.3.7 Initialization of Statistics

Statistics registers are hardware-initialized to values as detailed in each particular register's description. No initialization of these registers through software is necessary.

37.4.3.8 GbE Line Rate Configuration Change

The GbE unit must be reconfigured when switching rates with the RGMII gasket. The following guidelines apply:

- Disable receiver and transmitter by setting the RCTL.EN and TCTL.EN bits to 0.
- Wait an appropriate time for any packet being received to finish.
- Write CTRL.SPEED, CTRL.FRCSPD, CTRL.FD, CTRL.FRCDPLX with the targeted interface parameters.
- Re-enable receiver and transmitter by setting the RCTL.EN and TCTL.EN bits to 1.

37.4.3.9 Network Boot

Initialization of the GbE controller at power-up and/or reset from an optional serial EEPROM is supported. Refer to Section 37.5.11, “Serial EEPROM” on page 1412 for details on the use and configuration of the serial EEPROM.

Note: Section 22.2.4 of the IEEE 802.3 specification requires that PHY devices enter a normal operating state after power-up and/or reset without management intervention. As a result, it is believed that the MAC is not required to perform any sort of PHY initialization in the support of network boot.

37.4.3.10 Diagnostics

To assist in test and debug of device-driver software, a set of software-usable features have been provided. These features include controls for specific test-mode usage, as well as some registers for verifying device internal state against what the device-driver might be expecting.

The GbE provides software visibility (and controllability) into certain major internal data structures, including all of the transmit & receive FIFO space. However, interlocks are not provided for any operations, so diagnostic accesses should only be performed under very controlled circumstances.

The device also provides software-controllable support for certain loopback modes, to allow a device-driver to test transmit and receive flows to itself. Loopback modes may also be used to diagnose communication problems and attempt to isolate the location of a break in the communications path.

37.4.3.10.1 FIFO Pointer Accessibility

The internal pointers into the transmit and receive data FIFOs are visible through the head and tail diagnostic data FIFO registers. Diagnostic software may read these FIFO pointers to confirm an expected hardware state following a sequence of operations. Diagnostic software may further write to these pointers as a partial-step to verify expected FIFO contents following specific operation, or to subsequently write data directly to the data FIFOs.

37.4.3.10.2 FIFO Data Accessibility

The internal transmit and receive data FIFO contents are accessible through the Packet Buffer Memory (64KB) (PBM[n]) registers. The specific locations read/written are determined by the values of the FIFO pointers, which may also be read/written. When accessing the actual FIFO data structures, locations must be accessed as 32-bit words.

37.4.3.10.3 Loopback Operations

Loopback transmit/receive operation is a recommended debug tool. There are two points where the transmit data can be looped back to the receive path: in the PHY and in the MAC.

It is highly recommended that the PHY selected to interface with the GbE has implemented loopback operations in order to assist with system and device debug. Loopback operation may be used to test transmit & receive aspects of software drivers, as well as verify electrical integrity of the connections between the GbE and the system.

All loopback modes are only allowed when the MAC and PHY are configured for full duplex operation.

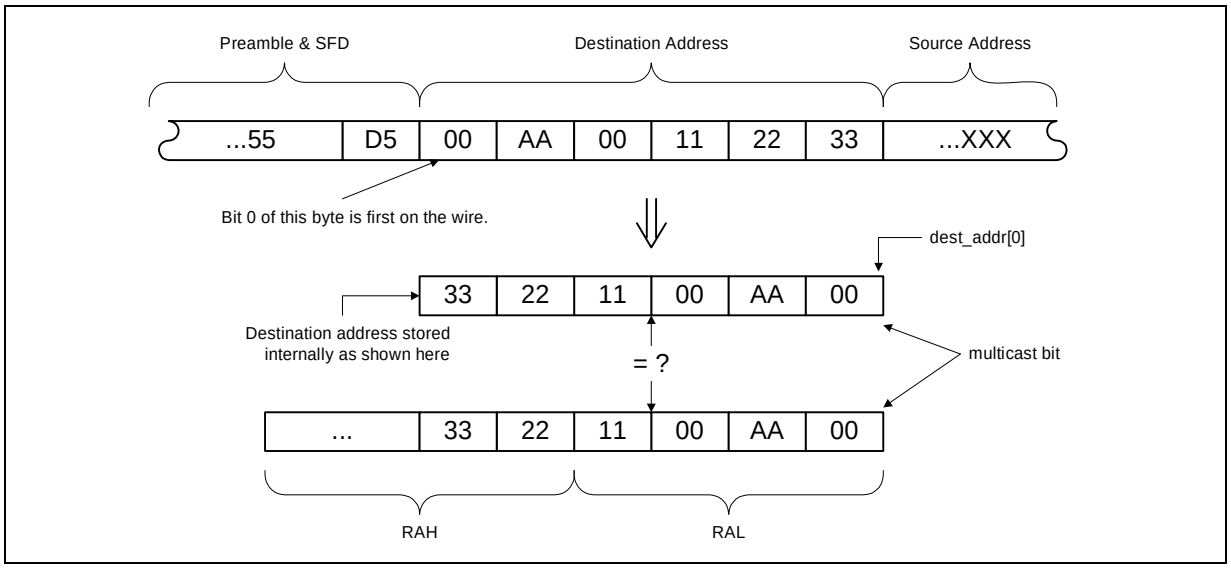
37.5 Functional Description

37.5.1 Ethernet Addressing

Several registers store Ethernet addresses in the GbE. Two 32-bit registers make up the address, the “high” and the “low”. For example, the Receive Address Register is comprised of Receive Address High (RAH) and Receive Address Low (RAL). The least significant bit of the least significant byte of the address stored in the register (i.e., bit 0 of RAL) is the multicast bit. The LS byte is the first byte to appear on the wire. This notation applies to all Ethernet address registers including flow control.

Figure 37-4 illustrates the bit/byte addressing order comparison between what is on the wire and the values in the unique receive address registers.

Figure 37-4. Example Address Byte Ordering



The address byte order numbering from the above example maps as shown below in Figure 37-5. Byte #1 is the first on the wire.

Figure 37-5. DA Byte Ordering

IA Byte Number	1(lsB)	2	3	4	5	6(msB)
Byte Value (HEX)	00	AA	00	11	22	33

Note: The notation in this document follows the above convention. For example, the above address would be indicated as 0x00_AA_00_11_22_33, where the first byte (00_) is the first byte on the wire with bit 0 of that byte transmitted first.

37.5.2 Interrupt Control & Tuning

The GbE provides a complete set of interrupts for efficient management by software. The interrupt structure is carefully designed to:

- Make accesses thread safe by using set and clear-on-read rather read-modify-write operations.
- Minimize the interrupts per work accomplished.
- Minimize the processing overhead associated with each interrupt.

The interrupt logic consists of four interrupt registers that accomplish the first goal. These registers are described in more detail in Section 37.6.3, “Interrupt Registers: Detailed Descriptions” on page 1454.

Three actions minimize the number of interrupts: reducing the frequency of all interrupts, accepting multiple receive packets before signaling an interrupt, and by eliminating (or reducing) the need for interrupts on transmit. Refer to “Interrupt Throttling Register” on page 1353, “Receive Interrupts” on page 1360, and “Transmit Interrupts” on page 1378 for details.

37.5.2.1 Interrupt Cause Set/Read Registers

The read register records the cause of the interrupt. All bits set at the time of the read are auto-cleared. The cause bit is set for each bit written as one in the set register. If there is a race between hardware setting a cause and software clearing an interrupt, the bit remains set. No race condition exists on writing the set register. Set provides for software posting of an interrupt. Reads are auto-cleared to avoid expensive write operations. Most systems have write buffering that minimizes overhead, but this typically requires a read operation to guarantee that the write has been flushed from posted buffers. Without auto-clear, the cost of clearing an interrupt can be as high as two reads and one write.

37.5.2.2 Interrupt Mask Set (Read)/Clear Registers

Interrupts appear only if the interrupt cause bit is a one and the corresponding interrupt mask bit is a one. Software blocks assertion of the interrupt wire by clearing the bit in the mask register. The cause bit stores the interrupt event regardless of the state of the mask bit. Clear and set make this register more “thread safe” by avoiding a read-modify-write operation on the mask register. The mask bit is set for each bit written to a one in the set register and cleared for each bit written in the clear register. Reading the set register returns the current value.

37.5.2.3 Interrupt Throttling Register

The frequency of functional interrupts from the network controller can be reduced when inter-interrupt interval value is non-zero. Controller asserts pending interrupts only at regularly scheduled intervals. When inter-interrupt interval value is zero, controller asserts pending interrupts immediately.

The Interrupt throttling register only applies to the GbE functional interrupt 0.

37.5.3 Hardware Acceleration Capability

The GbE provides CPU off loading capabilities. The functionality provided by these features may significantly reduce CPU utilization by shifting the burden of the functions from the driver to the hardware.

These features include:

- Receive & Transmit Checksum Off loading
- TCP Segmentation

These functions are covered in more detail in “Receive Packet Checksum Off loading” on page 1363, “Transmit Checksum Off loading” on page 1379, and “TCP Segmentation” on page 1380. The following sections provide a brief overview of these capabilities.

37.5.3.1 Checksum Off-Loading

The GbE provides the ability to off load the IPv4, TCP & UDP checksum requirements from the software device driver. For common frame types, the hardware automatically calculates, inserts and checks the appropriate checksum values typically handled by software.

Note: IPv6 headers do not have a checksum.

For transmits where the device is doing non-TCP segmentation, every transmitted Ethernet packet may have two checksums calculated and inserted by the device. Typically these would be the IPv4 and either TCP or UDP checksums. The driver

specifies which portions of the packet are included in the checksum calculations, and where the calculated values are inserted, via descriptor(s). See "TCP/IP Context Transmit Descriptor Format" on page 1369 for details.

For receives, the hardware recognizes the packet type and performs the checksum calculations as well as error checking automatically. Checksum and error information is provided to software via the receive descriptor(s). Refer to "Receive Packet Checksum Off loading" on page 1363 for details.

37.5.3.2 TCP Segmentation

The GbE implements a TCP segmentation capability for transmits which allows the software device driver to off load packet segmentation and encapsulation to the hardware. The device driver may send the GbE the entire IP (IPv4 or IPv6), TCP, or UDP message sent down by the NOS for transmission. The device will segment the packet into legal Ethernet frames and transmit them on the wire. By handling the segmentation tasks, the hardware alleviates the software from handling some of the framing responsibilities. This reduces the overhead on the CPU for the transmission process thus reducing overall CPU utilization. See "TCP Segmentation" on page 1380 for details.

37.5.4 Buffer and Descriptor Structure

Software allocates transmit and receive buffers and forms descriptors that contain pointers to and status of those buffers. A conceptual ownership boundary exists between the driver software and the hardware for buffers and descriptors.

Software gives hardware ownership of a queue of buffers for receive. These buffers store data that software acquires ownership of once a valid packet arrives.

For transmit, software maintains a queue of buffers. The software "owns" a buffer until it is ready to transmit. Software commits the buffer to the hardware at which time the hardware "owns" the buffer until data is transmitted or loaded in the transmit FIFO.

Descriptors store information about the buffers. They contain the physical address, length, and status information about the referenced buffer. An end-of-packet field indicates the last buffer for a packet.

The descriptors also contain packet specific information indicating type of packet and specific operations to perform in the context of transmitting a packet such as those for VLAN or checksum off load support. The following sections describe descriptor structure and operation in more detail in the context of packet transmission and reception.

37.5.5 Packet Reception

In the general case, packet reception consists of recognizing the presence of a packet on the wire, performing address filtering, storing the packet in the receive data FIFO, transferring the data to the receive buffer in host memory, updating the state of a receive descriptor, and setting the interrupt cause register to pass ownership of the received packet information to software.

37.5.5.1 Packet Address Filtering

Hardware stores incoming packets in host memory subject to the following filter modes. If there is insufficient space in the receive FIFO for an incoming packet, hardware drops the packet and indicates the missed packet in the appropriate statistics registers.

The following filter modes are supported:

Exact Unicast / Multicast - The destination address must exactly match one of 16 stored addresses. These addresses can be unicast or multicast.

Promiscuous Unicast - Receive all unicasts.

Multicast - The upper bits of the incoming packet's destination address index a bit vector that indicates whether to accept the packet; if the bit in the vector is one, accept the packet, otherwise, reject it. The EP80579's GbE provides a 4096 bit vector. Software provides four choices of which bits are used for indexing. These are [47:36], [46:35], [45:34], or [43:32] of the internally stored representation of the destination address.

Promiscuous multicast - Receive all multicast packets.

VLAN - Receive all VLAN packets that are for this station and have the appropriate bit set in the VLAN filter table. A detailed discussion and explanation of VLAN packet filtering is contained in "802.1q VLAN Packet Filtering" on page 1401.

Normally, only good packets are received. These are defined as those packets with no CRC error, symbol error, sequence error, length error, alignment error, or where carrier extension or RX_ERR errors are detected. However, if the store-bad-packet bit is set in the Device Control register (RCTL.SBP), then bad packets that pass the filter function are stored in host memory. Packet errors are indicated by error bits in the receive descriptor (RDESC.ERRORS). It is possible to receive all packets, regardless of whether they are bad, by setting the promiscuous enables and the store-bad-packet bit.

37.5.5.2 Receive Data Storage

Memory buffers pointed to by descriptors store packet data. Hardware supports various receive buffer sizes; explicitly 256B, 512B, 1024B, 2048B, 4096B, 8192B, and 16384B.

Buffer size is selected by bit settings in the Receive Control register (RCTL.BSIZE & RCTL.BSEX). See Section 37.6.4.1, "RCTL – Receive Control Register" for details.

The EP80579's GbE places no alignment restrictions on receive memory buffer addresses. This is desirable in situations where the receive buffer was allocated by higher layers in the networking software stack, as these higher layers may have no knowledge of a specific device's buffer alignment requirements.

Although alignment is completely unrestricted, it is highly recommended that software allocate receive buffers on cache-line boundaries.

37.5.5.3 Receive Descriptor Format

A receive descriptor is a data structure that contains the receive data buffer address and fields for hardware to store packet information. Refer to Figure 37-6, where the shaded areas indicate fields that are modified by hardware upon packet reception.

Figure 37-6. Receive Descriptor (RDESC) Layout

	63	48	47	40	39	32	31	16	15	0
0	Buffer Address[63:0]									
8	Special		Errors		Status		Packet Checksum†		Length	

† The checksum indicated here is the unadjusted "16 bit ones complement" of the packet. A software assist may be required to back out appropriate information prior to sending it up to upper software layers. The packet checksum is always reported in the first descriptor (even in the case of multi-descriptor packets).

† The Packet Checksum is reported in Little Endian format

Upon receipt of a packet for this device, hardware stores the packet data into the indicated buffer and writes the length, packet checksum, status, errors, and status fields. Length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for packets that span multiple receive buffers.

For standard 802.3 packets (non-VLAN) the Packet Checksum is by default computed over the entire packet from the first byte of the DA through the last byte of the CRC, including the Ethernet and IP headers. Software may modify the starting offset for the packet checksum calculation, refer to Section 37.6.4.13, "RXCSUM – Receive Checksum Control Register" for details. To verify the TCP checksum using the Packet Checksum, software must adjust the Packet Checksum value to back out the bytes that are not part of the true TCP Checksum.

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. Refer to Figure 37-7 for the layout of the status field. Error status information is shown in Figure 37-8.

Figure 37-7. Receive Status (RDESC.Status) Layout

7	6	5	4	3	2	1	0
PIF	IPCS	TCPCS	Rsvd	VP	IXSM	EOP	DD

PIF:	Passed in-exact filter
IPCS:	IPv4 Checksum Calculated on Packet
TCPCS:	TCP Checksum Calculated on Packet
Rsvd:	Reserved
VP:	Packet is 802.1q (matched VET MMR)
IXSM:	Ignore Checksum Indication
EOP:	End of Packet
DD:	Descriptor Done

Packets that exceed the receive buffer size span multiple receive buffers. EOP indicates whether this is the last buffer for an incoming packet. DD indicates whether hardware is done with the descriptor. When set along with EOP, the received packet is complete in main memory. Software can determine buffer usage by setting the status byte to 0 before making the descriptor available to hardware, and checking it for non-zero content at a later time. For multi-descriptor packets, packet status is provided in the final descriptor of the packet (EOP set). If EOP is not set for a descriptor, only the Address, Length, and DD bits are valid.

The VP field indicates whether the incoming packet's type matches VET (i.e., if the packet is a VLAN (802.1q) type). It will be set if the packet type matches VET and CTRL.VME is set. For a further description of 802.1q VLANs please see "802.1q VLAN Support" on page 1400.

When the Ignore Checksum Indication bit is deasserted (IXSM = 0), the IPCS and TCPCS bits indicate whether the hardware performed the IPv4 or TCP/UDP checksum(s) on the received packet, respectively. Pass/Fail information regarding the checksum is indicated in the status bits as described below for IPE & TCPE. When IXSM = 1, software should ignore the IPCS and TCPCS bits.

Refer to Table 37-1, “Supported Receive Checksum Capabilities” on page 1363 for a description of supported packet types for receive checksum off loading. Unsupported packet types will either have the IXSM bit set, or they will not have the IPCS or TCPCS bits set. IPv6 packets will not have the IPCS bit set, but may have the TCPCS bit set if the TCP or UDP packet was recognized by the EP80579’s GbE.

Hardware supplies the PIF field to expedite software processing of packets. Software must examine any packet with PIF set to determine whether to accept the packet. If PIF is clear, then the packet is known to be for this station, so software need not look at the packet contents. Packets passing only the Multicast Vector will have PIF set.

Most error information appears only when the store-bad-packet bit (RCTL.SBP) is set and a bad packet is received. Refer to Figure 37-8 below for a definition of the possible errors and their bit positions.

Figure 37-8. Receive Errors (RDESC.ERRORS) Layout

7	6	5	4	3	2	1	0
RXE	IPE	TCPE	CXE	Rsvd	Rsvd	Rsvd	CE

RXE:	RX Data Error
IPE:	IPv4 Checksum Error
TCPE:	TCP/UDP Checksum Error
CXE:	Carrier Extension Error (Reserved)
Rsvd:	Reserved
Rsvd:	Reserved
Rsvd:	Reserved
CE:	CRC Error or Alignment Error

The IP and TCP checksum error bits from Figure 37-8 are valid only when the IPv4 or TCP/UDP checksum(s) is performed on the received packet as indicated via IPCS and TCPCS. These, along with the other error bits, are valid only when the EOP and DD bit are set in the descriptor.

Note: Receive checksum errors have no affect on packet filtering.

If receive checksum off loading is disabled (RXCSUM.IPOFL & RXCSUM.TUOFL), the IPE and TCPE bits will be 0.

In GMII/MII mode, the RXE bit indicates that a data error occurred during the packet reception that has been detected by the PHY. This generally corresponds to signal errors occurring during the packet reception. This bit is valid only when the EOP and DD bits are set and will not be set in descriptors unless RCTL.SBP (store-bad-packets) is set.

CRC and alignment errors are indicated via the CE bit. Software may distinguish between these errors by monitoring the respective statistics registers.

Hardware stores additional information in the receive descriptor for 802.1q packets. If the packet type is 802.1q (determined when a packet matches VET and RCTL.VME = 1), then the special field records the VLAN information and the four byte VLAN information is stripped from the packet data storage. Otherwise, the special field contains 0x0000.

Figure 37-9. Special Descriptor Field Layout

802.1q Packets							
15 thru 8			7 thru 5		4	3 thru 0	
VLAN[7..0]			PRI		CFI	VLAN[11:8]	
All Other Packets							
15			8	7			0
00				00			

37.5.5.4 Receive Descriptor Fetching

The descriptor fetching strategy has been designed for the EP80579's GbE to support larger bursts across the internal bus. This is made possible by increasing the number of GbE hardware receive descriptors (from 8 to 64), and by modifying the fetch algorithm. The algorithm attempts to make the best use of the internal bus by fetching a cache-line (or more) of descriptors with each burst. The following paragraphs briefly describe the descriptor fetch algorithm and the software control provided.

When the descriptor buffer is empty, a fetch will happen as soon as any descriptors are made available (host writes to the tail pointer). When the descriptor buffer is nearly empty (as defined by RXDCTL.PTHRESH) a prefetch will be performed whenever enough valid descriptors (as defined by RXDCTL.HTHRESH) are available in host memory and no other internal bus activity of greater priority is pending (descriptor fetches, descriptor write-backs, or packet data transfers).

When the number of descriptors in host memory is greater than the available descriptor buffer storage, the GbE may elect to perform a fetch which is not a multiple of cache line size. The hardware performs this non-aligned fetch if doing so will result in the next descriptor fetch being aligned on a cache line boundary. This allows the descriptor fetch mechanism to be most efficient in the cases where it has fallen behind software.

Note: The GbE NEVER fetches descriptors beyond the descriptor TAIL pointer.

37.5.5.5 Receive Descriptor Write-Back

Processors have cache line sizes that are larger than the receive descriptor size (16 bytes). Consequently, writing back descriptor information for each received packet would cause expensive partial cache line updates. Two mechanisms minimize the occurrence of partial line write backs: receive descriptor packing and null descriptor padding.

37.5.5.5.1 Receive Descriptor Packing

To maximize memory efficiency, receive descriptors are "packed" together and written as a cache line whenever possible. Descriptors accumulate and are opportunistically written out in cacheline-oriented chunks. Used descriptors will also be explicitly written out under the following scenarios:

- RXDCTL.WTHRESH descriptors have been used (the specified max threshold of unwritten used descriptors has been reached)
- The last descriptors of the allocated descriptor ring have been used (to allow the hardware to re-align to the descriptor ring start)
- A receive timer expires (RADV or RDTR)

- Explicit software flush (RDTR.FPD)

When the numbers of descriptors specified by RXDCTL.WTHRESH have been used, they are written back, regardless of cacheline alignment It is therefore recommended that WTHRESH be a multiple of cacheline size. When a receive timer (RADV or RDTR) expires, all used descriptors are forced to be written back prior to initiating the interrupt, for consistency. Software may explicitly flush accumulated descriptors by writing the RDTR register with the high order bit (FPD) set.

37.5.5.5.2 Null Descriptor Padding

Hardware stores no data in descriptors with a null data address. Software can make use of this property to cause the first condition under receive descriptor packing to occur early. Hardware writes back null descriptors with the DD bit set in the status byte and all other bits unchanged.

37.5.5.6 Receive Descriptor Queue Structure

Figure 37-10.Receive Descriptor Ring Structure

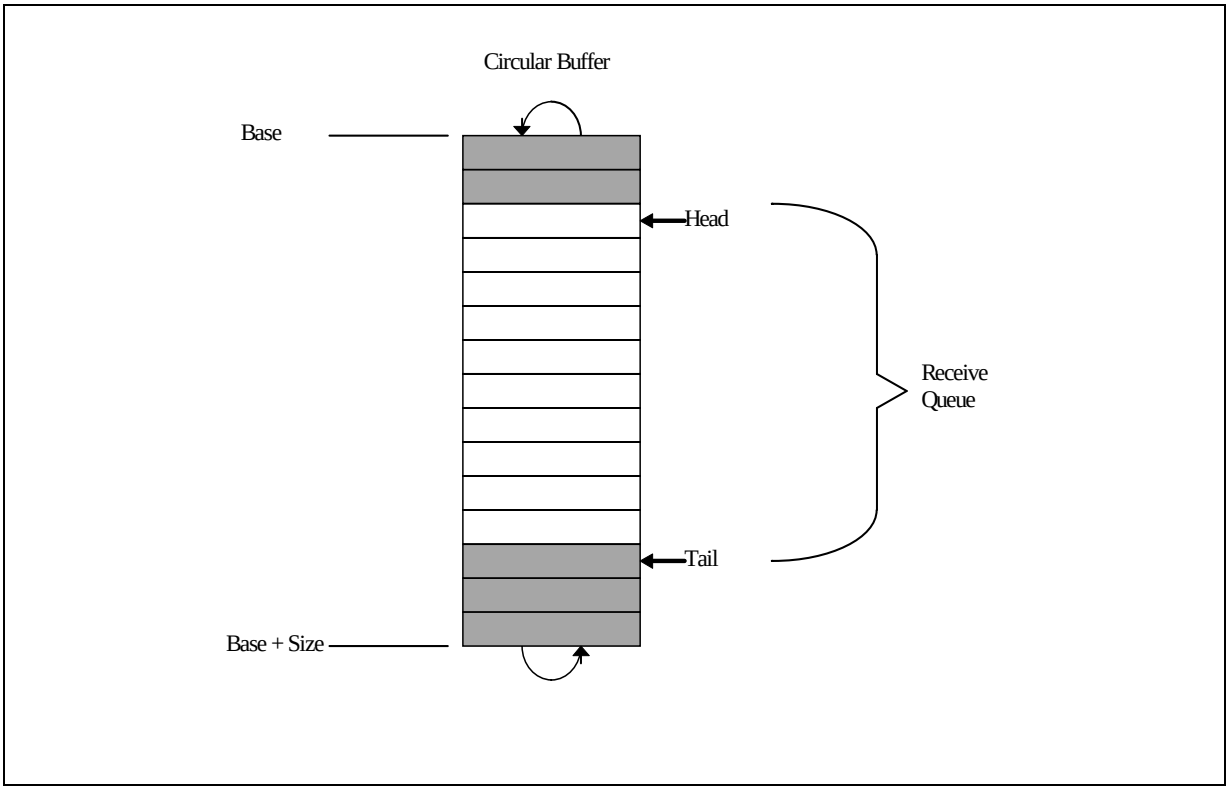


Figure 37-10 shows the structure of the receive descriptor ring. Hardware maintains a circular queue of descriptors and writes back used descriptors just prior to advancing the head pointer. Head and tail pointers wrap back to base when “size” descriptors have been processed.

Software adds receive descriptors by advancing the tail pointer to refer to the address of the entry just beyond the last valid descriptor. This is accomplished by writing the descriptor tail register with the offset of the entry beyond the last valid descriptor. The hardware adjusts its internal tail pointer accordingly. As packets arrive, they are stored

in memory and the head pointer is incremented by hardware. When the head pointer is equal to the tail pointer, the queue is empty. Hardware stops storing packets in system memory until software advances the tail pointer, making more receive buffers available.

The receive descriptor head and tail pointers reference 16-byte blocks of memory. Shaded boxes in the figure represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading descriptors in memory. Any descriptor with a non-zero status byte has been processed by the hardware, and is ready to be handled by the software.

Note: The head pointer points to the next descriptor that will be written back. At the completion of the descriptor write-back operation, this pointer is incremented by the number of descriptors written back. Hardware "owns" all descriptors between the head and tail. Any descriptor not in this range is owned by software.

The receive descriptor ring is described by the following registers:

The Receive Descriptor Base Address High Register and the Receive Descriptor Base Address Low Register (RDBAH and RDBAL) - These registers indicate the start of the descriptor ring buffer; this 64-bit address is aligned on a 16B boundary. Hardware ignores the lower 4 bits.

Receive Descriptor Length Register (RDLEN) - This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128. Since each descriptor is 16 bytes in length, the total number of receive descriptors is always a multiple of 8.

Receive Descriptor Head Register (RDH) - This register holds a value that is an offset from the base, and indicates the in-progress descriptor. There can be up to 64K descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory.

Receive Descriptor Tail Register (RDT) - This register holds a value that is an offset from the base, and identifies the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

If software statically allocates buffers, and uses memory read to check for completed descriptors, it simply has to zero the status byte in the descriptor to make it ready for reuse by hardware. This is not a hardware requirement (moving the hardware tail pointer is), but is necessary for performing an in-memory scan.

37.5.5.7 Receive Interrupts

The following sections indicate the presence of new packets.

37.5.5.7.1 Receive Timer (ICR.RXT0) Due to Absolute Timer (RADV)

- When a packet is received the Absolute Timer starts counting down. When it reaches 0 it generates an interrupt and resets itself. It is also reset if an interrupt is generated due to the Packet Delay Timer expiration. The absolute timer is disabled if RADV is 0. To use the Absolute Timer only the RDTR register must be set to an equal or greater value than RADV.

37.5.5.7.2 Receive Timer (ICR.RXT0) Due to Packet Delay Timer (RDTR)

- When a packet is received the Packet Delay Timer starts counting down. Every time an additional packet is received the Packet Delay Timer is reset to its starting value. When it reaches 0 it generates an interrupt and resets itself. It is also reset if an interrupt is generated due to the Absolute Timer expiration.
- When RDTR is 0 interrupts are immediate. If RADV is non-0, and RDTR is equal to or higher than RADV then the Packet Delay Timer will never generate any interrupts as the Absolute Timer will always generate them first.

- Writing RDTR with its high order bit 1 forces an explicit flush of any partial cache lines worth of consumed descriptors. Hardware writes all used descriptors to memory and updates the globally visible value of the RDH head pointer.

Figure 37-11 further describes the Packet Delay Timer operation, in general.

Figure 37-11.Packet Delay Timer operation illustrated with a state diagram

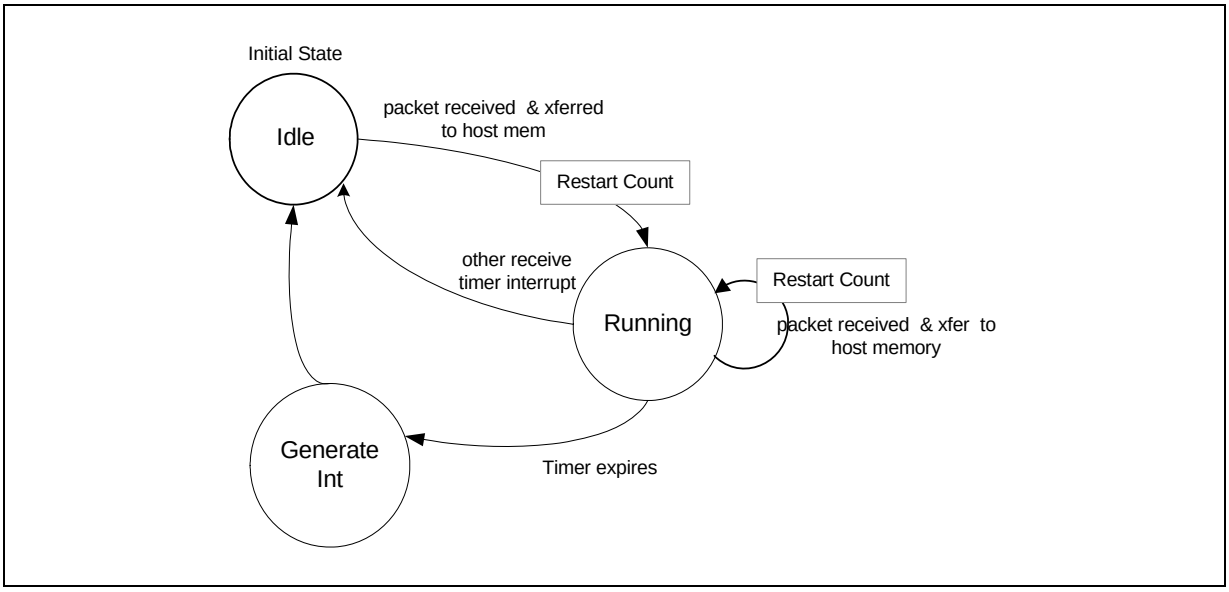


Figure 37-12, Figure 37-13, and Figure 37-14 illustrate the uses of the two timers.

Figure 37-12.Case A: Using only an Absolute Timer

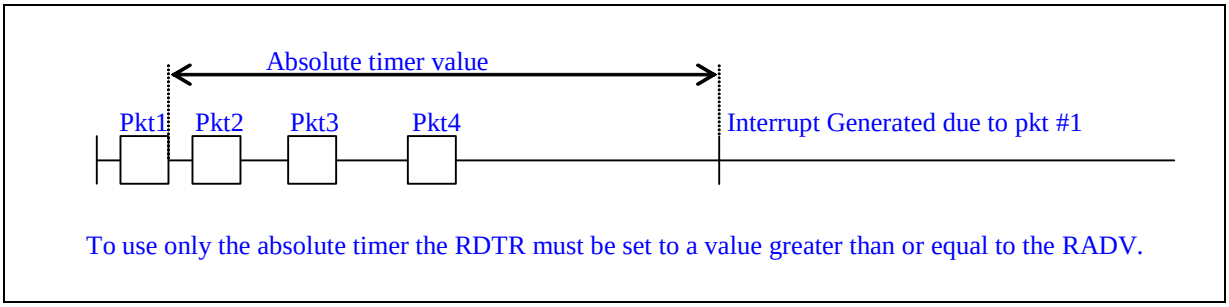
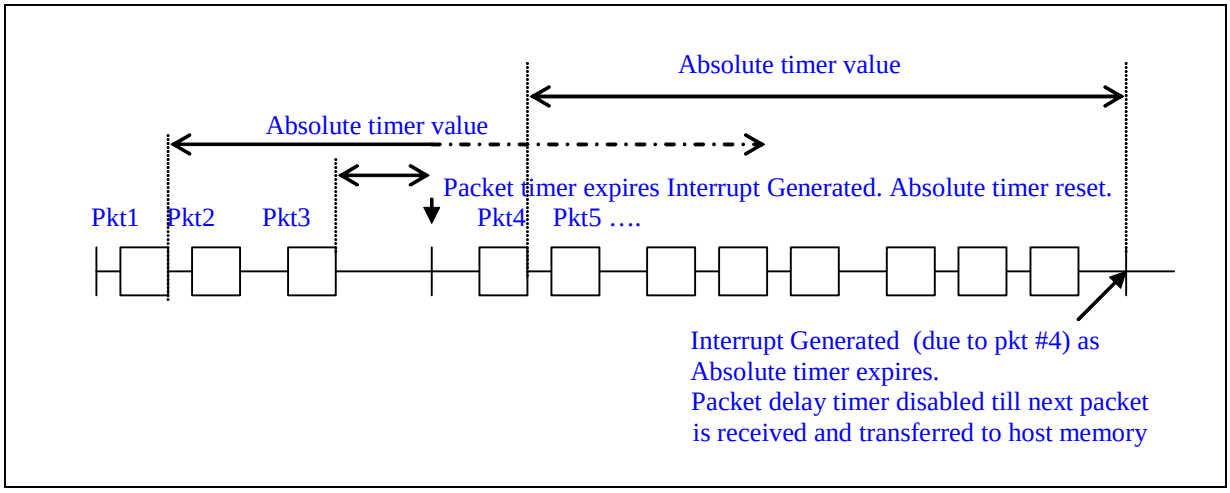
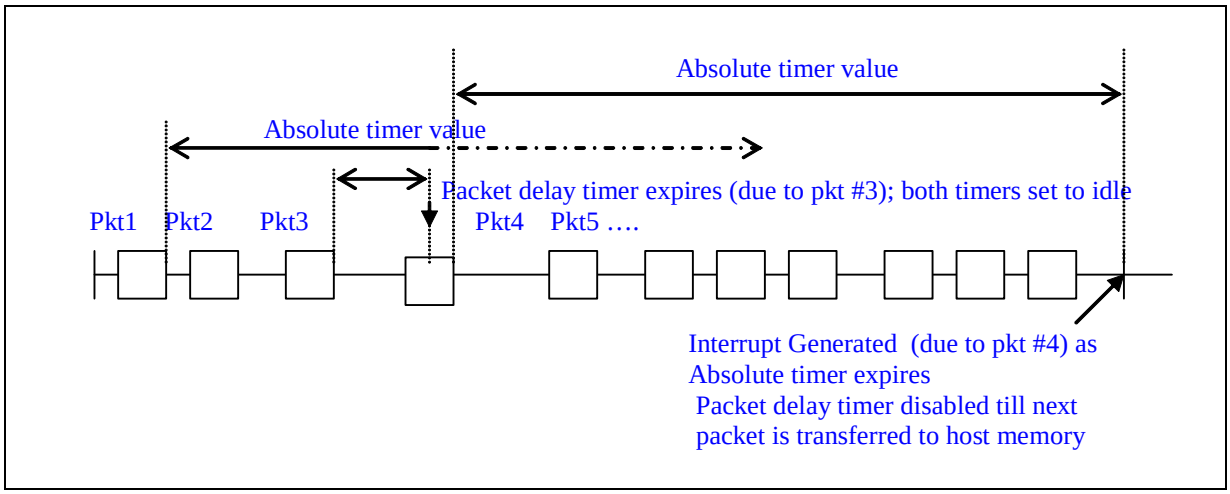


Figure 37-13.Case B: Using an Absolute Timer in conjunction with the Packet Timer



Case C in Figure 37-14 shows a scenario in which the packet timer expires even though a packet was being transferred to the host memory. Illustrating the fact that the packet timer is re-started only after a packet is transferred to host memory.

Figure 37-14.Case C: Packet Timer Expires Even Though A Packet Was Being Transferred to the Host Memory.



37.5.5.7.3 Small Receive Packet Detect (ICR.SRPD)

The Small Packet Receive Detect timer is independent of the other timers. It will generate a receive interrupt when small-packet detection is enabled (RSRPD is set with a non-zero value) and a packet of size less than or equal to RSRPD.SIZE has been transferred into the host memory. When comparing the size the headers and CRC are included (if CRC stripping is not enabled). CRC and VLAN headers are not included if they have been stripped.

Receiving a small packet does not clear the absolute or packet delay timers, so one packet may generate two receive interrupts, one due to the small packet reception and one due to a timer expiration.

37.5.5.7.4 Receive Descriptor Minimum Threshold (ICR.RXDMT)

The minimum descriptor threshold helps avoid descriptor under-run by generating an interrupt when the number of free descriptors becomes equal to the minimum. It is measured as a fraction of the receive descriptor ring size. This interrupt would stop and re-initialize all of the active delayed receives interrupt timers until a new packet is observed.

37.5.5.7.5 Receiver FIFO Overrun (ICR.RXO)

FIFO overrun occurs when hardware attempts to write a byte to a full FIFO. An overrun could indicate that software has not updated the tail pointer to provide enough descriptors/buffers, or that the internal bus is too slow draining the receive FIFO. Incoming packets that overrun the FIFO are dropped and do not affect future packet reception. This interrupt would stop and re-initialize all of the active delayed receive interrupts.

37.5.5.8 Receive Packet Checksum Off loading

The GbE supports the off loading of three receive checksum calculations: the Packet Checksum, the IPv4 Header Checksum, and the TCP/UDP Checksum. Ethernet II and Ethernet SNAP frame types are supported.

The Packet checksum is the one's complement of the receive packet, starting from the byte indicated by RXCSUM.PCSS (0 corresponds to the first byte of the packet), after stripping. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and with RXCSUM.PCSS set to 14, the Packet Checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, Type and Length) and the 4-byte q-tag. The Packet checksum will not include the Ethernet CRC if the RCTL.SECRC bit is set.

Software must make the required offsetting computation (to back out the bytes that should not have been included and to include the pseudo-header) prior to comparing the Packet Checksum against the TCP checksum stored in the packet.

For supported packet/frame types, the entire checksum calculation may be off-loaded to the GbE. If RXCSUM.IPOFLD is set to one, the GbE will calculate the IPv4 checksum and indicate a pass/fail indication to software via the IPv4 Checksum Error bit (RDESC.IPE) in the ERROR field of the receive descriptor. Similarly, if RXCSUM.TUOFLD is set to one, the GbE will calculate the TCP or UDP checksum and indicate a pass/fail condition to software via the TCP/UDP Checksum Error bit (RDESC.TCPE). These error bits are valid when the respective status bits indicate the checksum was calculated for the packet (RDESC.IPCS and RDESC.TCPCS respectively).

If neither RXCSUM.IPOFLD nor RXCSUM.TUOFLD are set, the Checksum Error bits (IPE & TCPE) will be 0 for all packets.

Table 37-1. Supported Receive Checksum Capabilities (Sheet 1 of 2)

Packet Type	HW IP Checksum Calculation	HW TCP/UDP Checksum Calculation
IPv4 packets	Yes	Yes
IPv6 packets	N/A	Yes
IPv6 packet with next header options:		
• Hop-by-Hop options	N/A	Yes
• Destinations options	N/A	Yes
• Routing	N/A	Yes
• Fragment	N/A	No

Table 37-1. Supported Receive Checksum Capabilities (Sheet 2 of 2)

IPv4 tunnels:		
• IPv4 packet in an IPv4 tunnel	No	No
• IPv6 packet in an IPv4 tunnel	Yes (IPv4)	Yes*
IPv6 tunnels:		
• IPv4 packet in an IPv6 tunnel	No	No
• IPv6 packet in an IPv6 tunnel	No	No
Packet is an IPv4 fragment	Yes	No
Packet is greater than 1552 bytes		
• LPE=1	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
IPv4 Packet has IP options		
• IP header is longer than 20 bytes	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains a protocol # other than TCP or UDP.	Yes	No

* The IPv6 header portion can include supported extension headers as described in the IPv6 Filter section.

Table 37-1 gives general details about what packets are processed. In more detail, the packets are passed through a series of filters to determine if a receive checksum is calculated.

37.5.5.8.1 MAC Address Filter

This filter checks the MAC destination address to be sure it is valid (e.g. DA match, broadcast, multicast, etc.). The receive configuration settings determine which MAC addresses are accepted. See the various receive control configuration registers such as RCTL (RCTL.UPE, RCTL.MPE, RCTL.BAM), MTA[127:0], RAL, and RAH.

37.5.5.8.2 SNAP/VLAN Filter

This filter checks the next headers looking for an IP header. It is capable of decoding Ethernet II, Ethernet SNAP, and IEEE 802.3ac headers. It will skip past any of these intermediate headers and will look for the IP header. The receive configuration settings determine which next headers are accepted. See the various receive control configuration registers such as RCTL (RCTL.VFE), VET, and VFTA[127:0].

37.5.5.8.3 IPv4 Filter

This filter checks for valid IPv4 headers. The version field is checked for a correct value (i.e. 4). IPv4 headers are accepted if they are any size greater than or equal to 5 (dwords). If the IPv4 header is properly decoded, the IP checksum will be checked for validity. The RXCSUM.IPOFL bit must be set for this filter to pass.

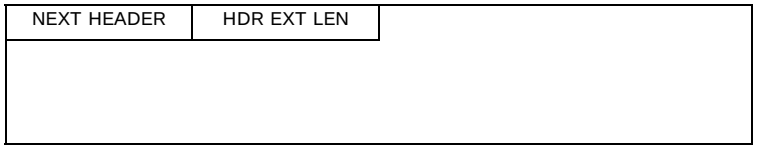
37.5.5.8.4 IPv6 Filter

This filter checks for valid IPv6 headers, which are a fixed size and have no checksum. The IPv6 extension headers accepted are: Hop-by-Hop, Destination Options, and Routing. The maximum size next header accepted is 16 dwords (64 bytes).

All of the IPv6 extension headers supported by the GbE have the same header structure, as shown in Figure 37-15.

Figure 37-15. IPv6 Extension Header Structure

Figure 37-15. IPv6 Extension Header Structure



NEXT HEADER is a value that identifies the header type. The supported IPv6 next headers values are:

- Hop-by-Hop = 00h
- Destination Options = 3Ch
- Routing = 2Bh

HDR EXT LEN is the 8 byte count of the header length, not including the first 8 bytes. For example, a value of 3 means that the total header size including the NEXT HEADER and HDR EXT LEN fields is 32 bytes (8 + 3*8).

37.5.5.8.5 UDP/TCP Filter

This filter checks for a valid UDP or TCP header. The prototype next header values are 11h and 06h, respectively. The RXCSUM.TUOFL bit must be set for this filter to pass.

37.5.6 Packet Transmission

Output packets are made up of pointer-length pairs constituting a descriptor chain (so called descriptor based transmission). Software forms transmit packets by assembling the list of pointer-length pairs, storing this information in the transmit descriptor, and then updating the GbE hardware transmit tail pointer to the descriptor. The transmit descriptor and buffers are stored in host memory. Hardware typically transmits the packet only after it has completely fetched all packet data from host memory and deposited it into the TX portion of the packet buffer. This permits TCP or UDP checksum computation, and avoids problems with internal bus under-runs.

Another transmit feature of the GbE is TCP Segmentation. The hardware has the capability to perform packet segmentation on large data buffers off-loaded from the NOS. This feature is discussed in detail in "TCP Segmentation" on page 1380.

37.5.6.1 Transmit Data Storage

Data are stored in buffers pointed to by the descriptors. Alignment of data is on an arbitrary byte boundary with the maximum size per descriptor limited only to the maximum allowed packet size (16288 bytes[1]). A packet typically consists of two (or more) descriptors, one (or more) for the header and one for the actual data. Some software implementations copy the header(s) and packet data into one buffer and use only one descriptor per transmitted packet.

37.5.6.2 Transmit Descriptor Formats

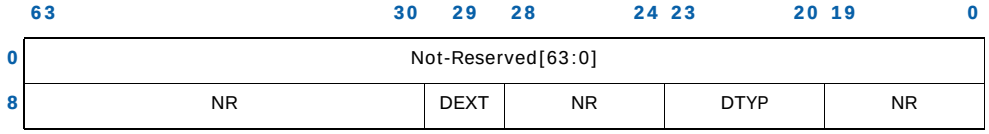
The EP80579's GbE has an expanded transmit descriptor format in that additional descriptor types have been introduced. The intent of the descriptor type field is to provide an extensible interface to the GbE descriptor mechanisms while maintaining a great deal of compatibility with the existing descriptor format that the 82542 used.

1. This value is based on the default packet buffer allocation of 16KB Transmit; 48KB Receive. The maximum allowable transmit packet size is the transmit allocation minus 96 bytes with an upper limit of 24KB due to receive synchronization limitations.

The original (82542 compatible) descriptor will be referred to as the “legacy” descriptor format and is described in “Legacy Transmit Descriptor Format” on page 1366. The two new descriptor types are collectively referred to as extended descriptors. One of the new descriptor types is quite similar to the legacy descriptor in that it points to a block of packet data. This descriptor type is called the TCP/IP Data Descriptor and is offered as a replacement for the legacy descriptor since it offers access to new off loading capabilities. The other new descriptor type is fundamentally different as it does not point to packet data. It merely contains control information which are loaded into registers of the GbE and affect the processing of future packets. The following paragraphs describe the three descriptor formats.

Note: The extended descriptor types are accessed by setting the TDESC.DEXT bit to 1. If this bit is set, the TDESC.DTYP field is examined to control the interpretation of the remaining bits of the descriptor. Figure 37-16 shows the generic layout for all extended descriptors. Fields marked as NR are not reserved for any particular function and are defined on a per-descriptor type basis. Notice that the DEXT and DTYP fields are non-contiguous in order to accommodate legacy mode operation. For legacy mode operation, bit 29 is set to 0 and the descriptor is defined as described in “Legacy Transmit Descriptor Format” on page 1366.

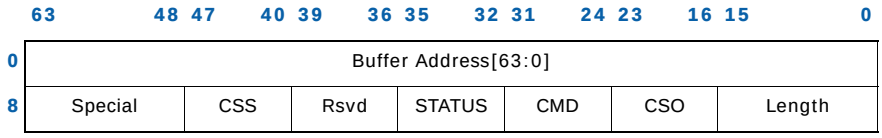
Figure 37-16.Transmit Descriptor (TDESC) Layout



37.5.6.3 Legacy Transmit Descriptor Format

To select legacy (82542 compatible) mode operation, bit 29 (TDESC.DEXT) should be set to 0. In this case, the descriptor format is defined as shown in Figure 37-17. Address and length must be supplied by software. Bits in the command byte are optional, as are the CSO, and CSS fields.

Figure 37-17.Legacy Transmit Descriptor (TDESC) Layout



Length (TDESC.LENGTH) specifies the length in bytes to be fetched from the buffer address provided. The maximum length associated with any single legacy descriptor is 16288 bytes. Although a buffer as short as one byte is allowed, the total length of the packet, before padding and CRC insertion, must be at least 17 bytes.

Note: The maximum allowable packet size for transmits changes based on the value written to the Packet Buffer Allocation Register, whose settings can modify the size of the transmit FIFO.

Descriptor length(s) may be further limited by the size of the transmit FIFO. Due to the need to support optional checksum calculation & insertion, all buffers comprising a single packet must be able to be stored simultaneously in the transmit FIFO. For any individual packet, the sum of the individual descriptor lengths must be at least 80 bytes less than the allocated size of the transmit FIFO.

A Checksum offset (TDESC.CSO) field indicates where to insert a TCP checksum if this mode is enabled, relative to the start of the packet. A Checksum start (TDESC.CSS) field indicates where to begin computing the checksum. Both CSO and CSS are in units of bytes. These must both be in the range of data provided to the device in the descriptor. This means for short packets which are padded by software, CSS and CSO must be in the range of the unpadded data length, not the eventual padded length (64 bytes). Hardware will not add the 802.1Q EtherType or the VLAN field following the 802.1Q EtherType to the checksum. So for VLAN packets, software can compute the values to back out only on the encapsulated packet rather than on the added fields.

Note: Although the EP80579's GbE can be programmed to calculate and insert TCP checksum using the legacy descriptor format as described above, it is recommended that software use the newer TCP/IP Context Transmit Descriptor Format. This newer descriptor format allows the hardware to calculate both the IP and TCP checksums for outgoing packets. Refer to "Transmit Checksum Off loading" on page 1379 for more information about how the new descriptor format can be used to accomplish this task.

The CMD byte stores the applicable command and has the fields shown in Figure 37-18.

Figure 37-18.Transmit Command (TDESC.CMD) Layout

7	6	5	4	3	2	1	0
IDE	VLE	DEXT	RPS	RS	IC	IFCS	EOP

IDE: Interrupt Delay Enable

VLE: VLAN Packet Enable

DEXT: Descriptor Extension (0 for legacy mode)

RPS: Report Packet Sent

RS: Report Status

IC: Insert Checksum

IFCS: Insert FCS

EOP: End of Packet

EOP indicates the last descriptor making up the packet when asserted. One or many descriptors can be used to form a packet. Hardware inserts a checksum at the offset indicated by the CSO field if the Insert Checksum bit (IC) is set. Checksum calculations are for the entire packet starting at the byte indicated by the CSS field. A value of 0 corresponds to the first byte in the packet. Hardware ignores IC, and CSO unless EOP is set. CSS must be set in the first descriptor for a packet. In addition, IC is ignored if CSO or CSS are out of range. This occurs if (CSS >= length) OR (CSO >= length - 1).

Software must compute an offsetting entry-to back out the bytes of the header that should not be included in the TCP checksum-and store it in the position where the hardware computed checksum is to be inserted.

TDESC.CMD.RS tells the hardware to report the status information. This is used by software that does in-memory checks of the transmit descriptors to determine which ones are done. For example, if software queues up 10 packets to transmit, it can set the RS bit in the last descriptor of the last packet. If software maintains a list of descriptors with the RS bit set, it can look at them to determine if all packets up to (and including) the one with the RS bit set have been buffered in the output FIFO. This is done by looking at the status byte and checking the Descriptor Done (DD) bit. If DD is set, the descriptor has been processed. Refer to Figure 37-19 on page 1368 for the layout of the status field.

Note: Descriptors with the null address (0), or zero length, transfer no data. If they have the RS bit in the command byte set, then the DD field in the status word is written after hardware processes them. Hardware only sets the DD bit for descriptors with RS set.

Note: Null Descriptors are intended for padding descriptor queues, in case a specific alignment of descriptors comprising a packet are desired. Null transmit descriptors should not be used to convey any meaningful command information (such as EOP); they are consumed with no processing other than status reporting (if requested).

Note: Hardware is considered “done processing” a descriptor when any data specified in the descriptor has been completely fetched and loaded in the transmit FIFO.

Note: VLE, IFCS, and IC are qualified by EOP. In other words, hardware interprets these bits ONLY when EOP is set.

IDE activates a transmit interrupt delay timer. Hardware loads a countdown register when it writes back a transmit descriptor that has RS and IDE set. The value loaded comes from Transmit Interrupt Delay Value Register (TIDV.IDV). When the count reaches 0, a transmit interrupt occurs if transmit descriptor write-back interrupts (ICR.TXDW) are enabled. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor and the interrupt delay timer is cleared.

Note: Although the transmit interrupt may be delayed, the descriptor write-back requested by setting the RS bit is performed without delay unless descriptor write-back bursting is enabled. See Section 37.6.5.10, “TXDCTL – Transmit Descriptor Control Register”.

VLE indicates that the packet is a VLAN or ISL packet (i.e. that the hardware should add the VLAN Ethertype and an 802.1q VLAN tag to the packet).

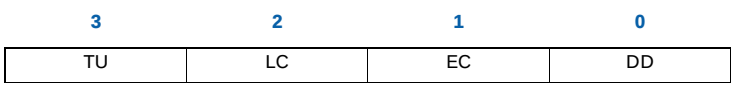
Note: If the VLE bit is set, the CTRL.VME bit should also be set to enable VLAN tag insertion. If the CTRL.VME bit is not set, the device will not insert VLAN tags on outgoing packets, but may instead insert ISL headers.

Table 37-2. VLAN Tag Insertion Decision Table when VLAN Mode Enabled (CTRL.VME=1)

VLE	Action
0	Send generic Ethernet packet. IFCS controls insertion of FCS in normal Ethernet packets.
1	Send 802.1Q packet; the Ethernet Type field comes from the VET register and the VLAN data comes from the special field of the TX descriptor; hardware always appends the FCS/CRC.

Three bits provide transmit status. These are only present in cases where RS is set in the command. DD indicates that the descriptor is done and is written back after the descriptor has been processed. The bits EC and LC indicate collision behavior when in half-duplex mode. They have no meaning when in full-duplex mode.

Figure 37-19.Transmit Status Layout (TDESC.STATUS)



TU: Transmit Underrun

LC: Late Collision

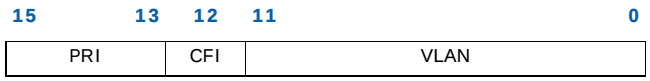
EC: Excess Collisions

Figure 37-19.Transmit Status Layout (TDESC.STATUS)

DD: Descriptor Done

The SPECIAL field is used to provide the 802.1q/802.1ac tagging information. The special field is ignored if the VLE bit is 0 or if the EOP bit is 0.

Figure 37-20.Transmit Special Field Layout (TDESC.SPECIAL)



37.5.6.4 TCP/IP Context Transmit Descriptor Format

The TCP/IP context transmit descriptor provides access to the enhanced checksum off load and TCP Segmentation facilities available in the GbE. These features allow TCP and UDP packet types to be handled more efficiently by performing additional work in hardware, thus reducing the software overhead associated with preparing these packets for transmission.

The TCP/IP context transmit descriptor is called a “context” descriptor for a reason. A context descriptor differs from a data descriptor as it does not point to packet data. Instead, this descriptor provides access to two, GbE hardware contexts that support the transmit checksum off loading and the segmentation features of the EP80579’s GbE. A “context” refers to a set of registers loaded or unloaded as a group to provide a particular function. Only one context is active at any given time.

37.5.6.4.1 Overview of GbE hardware “Contexts”

The previous section mentions access to two, separate GbE hardware contexts. There are actually three such contexts corresponding to legacy mode, “normal” mode, and segmentation mode. The first context (legacy) is an implied context as it is not explicitly specified with a context descriptor. This context is constructed by the device from the first and last descriptors of a legacy transmit and from some internal constants. This context then mimics the behavior of the legacy (82542) device. This is completely transparent to the user and is included here for clarification only.

The other two contexts are explicit and directly accessible via the TCP/IP context transmit descriptor. One context is used to control the checksum off loading feature for normal packet transmission. The second context is used to control the packet segmentation capabilities of the device. The TSE bit selects which context will be updated. A TCP/IP context transmit descriptor with TSE=0 will update the normal (checksum off loading only) context. Conversely, the segmentation context is updated when TSE=1. Refer to Figure 37-22 on page 1371 for details on all of the TDESC.TUCMD fields, including TSE.

The device automatically selects the appropriate context to use based on the current packet transmission: legacy, normal, or segmentation.

Note: While the architecture supports arbitrary ordering rules for the various descriptors, there are restrictions. The context descriptors should not occur in the middle of a packet (or of a segmentation) and data descriptors of different packet types (legacy, normal, or segmentation) should not be intermingled except at the packet (or segmentation) level.

All three contexts control calculation and insertion of up to two checksums. This portion of the context is referred to as the checksum context. In addition to a checksum context, the segmentation context includes information specific to the segmentation capability. This additional information includes the total payload for the message (TDESC.PAYLEN), the total size of the header (TDESC.HDRLEN), the amount of payload

data that should be included in each packet (TDESC.MSS), and information about what type of protocol (TCP, IPv4, IPv6, etc.) is used. This information is specific to the segmentation capability and is therefore ignored for context descriptors that do not have the TSE bit set.

Due to the fact that there are dedicated resources in the GbE hardware for both the normal and segmentation contexts, these contexts will remain constant until they are modified by another context descriptor. This means that a context can be used for multiple packets unless a new context is loaded prior to each new packet. Depending on the environment, it may be completely unnecessary to load a new context for each packet. For example, if most traffic generated from a given node is standard TCP frames, this context could be setup once and used for many frames. Only when some other frame type is required would a new context need to be loaded by software. After the “non-standard” frame is transmitted, the “standard” context would be setup once more by software. This method avoids the “extra descriptor per packet” penalty for most frames. The penalty can be eliminated altogether if software elects to use TCP/IP checksum off loading only for a single frame type, and thus performing those operations in software for other frame types.

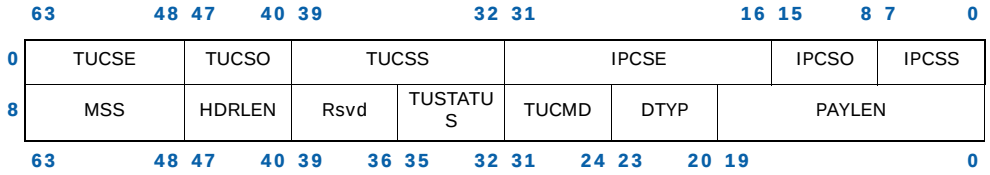
A segmentation context similarly describes parameters which are setup once and used for many frames. Since a segmentation operation results in the generation of multiple frames, the context parameters such as checksum generation are applied to all frames generated. Unlike the “normal” TCP/IP context, a segmentation context cannot be re-used for multiple TCP messages, however, a new segmentation context is required for each segmentation operation.

37.5.6.4.2 TCP/IP Context descriptor layout

Note: The TCP/IP context transmit descriptor does not transfer any packet data. It merely prepares the checksum and segmentation hardware for the TCP/IP Data descriptors that follow.

To select the TCP/IP context transmit descriptor format, shown below in Figure 37-21, bit 29 (TDESC.DEXT) must be set to 1 and TDESC.DTYP must be set to “0000”.

Figure 37-21.TCP/IP Context Transmit Descriptor (TDESC) - (Type = 0000)



The first QWORD of this descriptor type contains parameters used to calculate the two checksums which may be off-loaded.

TDESC.IPCSS, TDESC.IPCSO, and TDESC.IPCSE specify the start, offset, and ending byte for the IP checksum off-load feature. Similarly, TDESC.TUCSS, TDESC.TUCSO, TDESC.TUCSE specify the same parameters for the TCP or UDP checksum. Setting either CSE field to 0 indicates the given checksum will cover from CSS to the end of packet. While the names imply particular packet types, the mechanisms are generic.

Note: When setting the TCP Segmentation context, IPCSS and TUCSS are used to indicate the start of the IP and TCP headers respectively, and must be set even if checksum insertion is not desired.

In certain situations, software may need to calculate a partial checksum (the TCP pseudo-header for instance) to include bytes which are not contained within the range of start and end. If this is the case, this partial checksum should be placed in the

packet data buffer, at the appropriate offset for the checksum. If no partial checksum is required, software must write a value of zero at this offset. Refer to the "Transmit Checksum Off loading" on page 1379 for additional details.

The second QWORD of this descriptor primarily contains information to support the TCP Segmentation feature. A number of the fields are ignored if the TCP Segmentation enable bit indicates that the descriptor does not refer to the TCP Segmentation context (TSE=0).

MSS controls the Maximum Segment Size. This specifies the maximum TCP or UDP payload "segment" sent per frame, not including any header. The total length of each frame (or "section") sent by the TCP Segmentation mechanism (excluding 802.3ac tagging and Ethernet CRC) will be MSS bytes + HRDLEN. The one exception is the last packet of a TCP segmentation context which will (typically) be shorter than "MSS + HDRLEN". This field is ignored if TSE is not set.

HDRLEN is used to specify the length (in bytes) of the header to be used for each frame (or "section") of a TCP Segmentation operation. The first HDRLEN bytes fetched from data descriptor(s) will be stored internally and used as a prototype header for each section, and will be prepended to each payload segment to form individual frames. For UDP packets this will normally be equal to "UDP checksum offset + 2". For TCP packets it will normally be equal to "TCP checksum offset + 4 + TCP header option bytes". This field is ignored if TDESC.TSE is not set.

Maximum limits for the HDRLEN and MSS fields are dictated by the length variables. However, there is a further restriction that the for any TCP Segmentation operation, the hardware must be capable of storing a complete framed fragment (completely-built frames) in the transmit FIFO prior to transmission. Therefore, the size of the TX packet buffer must be greater than or equal to ((MSS + HDRLEN) x Number of Frames Formed for this fragment). The quantity (HDRLEN x Number of Frames) should be set by software in the TCP Segmentation Pad And Minimum Threshold Register (TSPMT.TSPBP) to prevent packet buffer overflow.

The packet payload length field (TDESC.PAYLEN) is the total number of payload bytes for this TCP Segmentation off load context (i.e. the total number of payload bytes that could be distributed across multiple frames after TCP segmentation is performed). Following the fetch of the prototype header, PAYLEN specifies the length of data that will be fetched next from data descriptor(s). This field is also used to determine when "last-frame" processing needs to be performed. Typically, a new data descriptor is used to denote the start of the payload data buffer(s), but this is not required. PAYLEN specification should not include any header bytes. There is no restriction on the overall PAYLEN specification with respect to the transmit FIFO size, once the MSS and HDRLEN specifications are legal. This field is ignored if TDESC.TSE is not set. Refer to "TCP Segmentation" on page 1380 for details on the TCP Segmentation off-loading feature.

Setting the descriptor type (TDESC.DTYP) field to 0000 identifies this descriptor as a TCP/IP context transmit descriptor.

The command field (TDESC.TUCMD) provides options that control the checksum off-loading and TCP Segmentation features, along with some of the generic descriptor processing functions. Figure 37-22 shows the bit definitions for the TDESC.TUCMD field. The IDE, DEXT, and RS bits are valid regardless of the state of TSE. All other bits are ignored if TSE=0.

Figure 37-22.TCP/IP Context Transmit Descriptor Command Field (TDESC.TUCMD)

7	6	5	4	3	2	1	0
IDE	Rsvd	DEXT	Rsvd	RS	TSE	IP	TCP

IDE: Interrupt Delay Enable

Figure 37-22.TCP/IP Context Transmit Descriptor Command Field (TDESC.TUCMD)

Rsvd: Reserved

DEXT: Descriptor Extension (Must be 1 for this descriptor type)

Rsvd: Reserved

RS: Report Status

TSE: TCP Segmentation Enable

IP: IP Packet Type (IPv4=1, IPv6=0)

TCP: Packet Type (TCP=1,UDP=0)

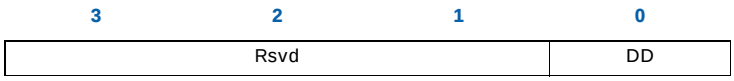
The TCP bit identifies the packet as either TCP or UDP (non-TCP). This effects the processing of the header information. The IP bit is used to indicate what type of IP packet is used in the segmentation process. This is necessary for the EP80579's GbE to know where the IP Payload Length field is located. This does not override the checksum insertion bit, TDESC.POPTS IXSM bit. The IP bit must only be set for IPv4 packets and cleared for IPv6 packets.

The TCP Segmentation feature also provides access to a generic block send function and may be useful for performing "segmentation offload" in which the header information is constant. By clearing both the TCP and IP bits, a block of data may be broken down into frames of a given size, a constant, arbitrary length header may be prepended to each frame, and two checksums optionally added.

TSE indicates that this descriptor is setting the TCP segmentation context. If this bit is not set, the checksum off loading context for normal (non-"TCP Segmentation") packets is written. When a descriptor of this type is processed, the device will immediately update the context in question (TCP Segmentation or checksum off loading) with values from the descriptor. This means that if any normal packets or TCP Segmentation packet are in progress (a descriptor with EOP set has not been received for the given context) the results will likely be undesirable.

RS tells the hardware to report the status information for this descriptor. Because this descriptor does not transmit data, only the DD bit in the status word will be valid. Refer to Figure 37-23 for the layout of the status field.

Figure 37-23.TCP/IP Context Transmit Descriptor Status (TDESC.TUSTATUS)



Rsvd: Reserved

DD: Descriptor Done

The reserved bits are reserved values and are ignored, but if written should be set to 0 for future compatibility.

The DEXT bit identifies this descriptor as one of the extended descriptor types and must be set to 1.

IDE activates the transmit interrupt delay timer. Hardware loads a countdown register when it writes back a transmit descriptor that has the RS bit and the IDE bit set. The value loaded comes from the Transmit Interrupt Delay Value Register (TIDV.IDV). When the count reaches 0, a transmit interrupt occurs. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already

counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor and the interrupt delay timer is cleared.

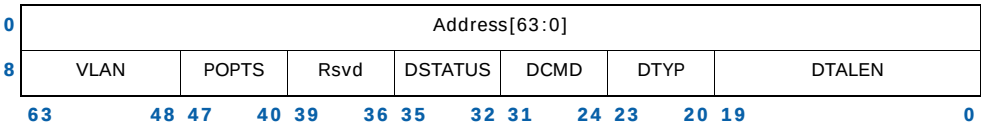
Four bits are reserved to provide transmit status, although only one is currently assigned for this specific descriptor type. The status word will only be written back to host memory in cases where the RS is set in the command. DD indicates that the descriptor is done and is written back after the descriptor has been processed.

37.5.6.5 TCP/IP Data Descriptor Format

The TCP/IP data descriptor is the companion to the TCP/IP context descriptor described in the previous section. This descriptor type provides similar functionality to the legacy mode descriptor but also integrates the checksum off loading and TCP Segmentation features.

To select the TCP/IP data transmit descriptor format, shown below in Figure 37-24.

Figure 37-24.TCP/IP Data Transmit Descriptor Layout (TDESC) - (Type = 0001)



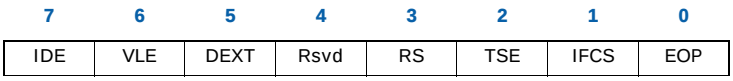
The first QWORD of this descriptor type contains the address of a data buffer in host memory which contains a portion of a transmit packet. The second QWORD of this descriptor contains information about the data pointed to by this descriptor as well as descriptor processing options.

Setting TDESC.DEXT to 1 and the descriptor type (TDESC.DTYP) field to “0001” identifies this descriptor as a TCP/IP data descriptor.

The data length field (TDESC.DTALEN) is the total length of the data pointed to by this descriptor, in bytes. For data descriptors not associated with a TCP Segmentation operation (TDESC.TSE not set), the descriptor lengths are subject to the same restrictions specified for legacy descriptors (the sum of the lengths of the data descriptors comprising a single packet must be at least 80 bytes less than the allocated size of the transmit FIFO). In addition, although a buffer as short as one byte is allowed, the total length of the packet, before padding and CRC insertion, must be at least 17 bytes.

The command field (TDESC.DCMD) provides options that control the checksum off-loading TCP Segmentation features, along with some of the generic descriptor processing features. Figure 37-25 shows the bit definitions for the DCMD field.

Figure 37-25.TCP/IP Data Transmit Descriptor Command Field (TDESC.DCMD)



IDE: Interrupt Delay Enable

VLE: VLAN Enable

DEXT: Descriptor Extension (Must be 1 for this descriptor type)

Figure 37-25.TCP/IP Data Transmit Descriptor Command Field (TDESC.DCMD)

Rsvd: Reserved (must be set to 0)

RS: Report Status

TSE: TCP Segmentation Enable

IFCS Insert FCS

EOP End of Packet

TSE indicates that this descriptor is part of the current TCP Segmentation context. If this bit is not set, the descriptor is part of the “normal” context.

A packet is sent from one or more data buffers in host memory. The EOP bit indicates that the buffer associated with this descriptor contains the last data for the packet or given TCP segmentation context. In the case of a TCP Segmentation context, the DTALEN length of this descriptor should match the amount remaining of the original PAYLEN. If it does not, the TCP Segmentation context will be terminated but the end of packet processing may be incorrectly performed. These abnormal termination events will be counted in the TSCTFC statistics register detailed in Section 37.6.6.53, “TSCTFC – TCP Segmentation Context Transmit Fail Count Register”.

IFCS controls insertion of the Ethernet CRC.

RS tells the hardware to report the status information for this descriptor as soon as the corresponding data buffer has been fetched and stored in the EP80579’s GbE internal packet buffer. Refer to Figure 37-26 for the layout of the status field.

The DEXT bit identifies this descriptor as one of the extended descriptor types and must be set to 1.

VLE indicates that the packet is a VLAN packet (i.e. that the hardware should add the VLAN Ethertype and an 802.1q VLAN tag to the packet).

Note: If the VLE bit to enable VLAN tag insertion, the CTRL.VME bit should also be set. If the CTRL. VME bit is not set, the device will not insert VLAN tags on outgoing packets.

Table 37-3. VLAN Tag Insertion Decision Table

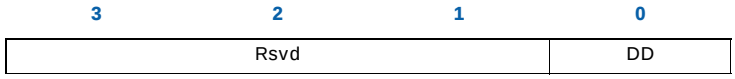
VLE	Action
0	Send generic Ethernet packet. IFCS controls insertion of FCS in normal Ethernet packets.
1	Send 802.1Q packet; the Ethernet Type field comes from the VET register and the VLAN data comes from the special field of the TX descriptor; hardware always appends the FCS/CRC.

Note: The VLE, IFCS, and VLAN fields are only are only valid in certain descriptors. If TSE is enabled, the VLE, IFCS, and VLAN fields are only valid in the first data descriptor of the TCP segmentation context. If TSE is not enabled, then these fields are only valid in the last descriptor of the given packet (qualified by the EOP bit).

IDE activates the transmit interrupt delay timer. Hardware loads a countdown register when it writes back a transmit descriptor that has RS and IDE set. The value loaded comes from Transmit Interrupt Delay Value Register (TIDV.IDV). When the count reaches 0, a transmit interrupt occurs. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor and the interrupt delay timer is cleared.

Four bits are reserved to provide transmit status, although only the DD bit (bit 0) is valid. The status word will only be written back to host memory in cases where the RS bit is set in the command field. The DD bit indicates that the descriptor is done and is written back after the descriptor has been processed.

Figure 37-26.TCP/IP Data Transmit Descriptor Status (TDESC.DSTATUS)

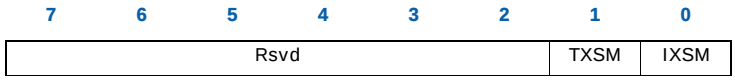


Rsvd: Reserved

DD: Descriptor Done

The POPTS field provides a number of options which control the handling of this packet. This field is ignored except on the first data descriptor of a packet or segmentation context.

Figure 37-27.TCP/IP Data Transmit Descriptor Packet Options Field (TDESC.POPTS)



Rsvd: Reserved

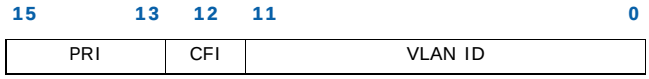
TXSM: Insert TCP/UDP Checksum

IXSM: Insert IP Checksum

IXSM and TXSM are used to control insertion of the IP and TCP/UDP checksums, respectively. If the corresponding bit is not set, whatever value software has placed into the checksum field of the packet data will be placed on the wire. These bits are only valid in the first data descriptor for a given packet or TCP segmentation context.

The VLAN field is used to provide the 802.1q tagging information. The special field is ignored if the VLE bit is 0.

Figure 37-28.TCP/IP Data Transmit Descriptor Special Field (TDESC.VLAN)



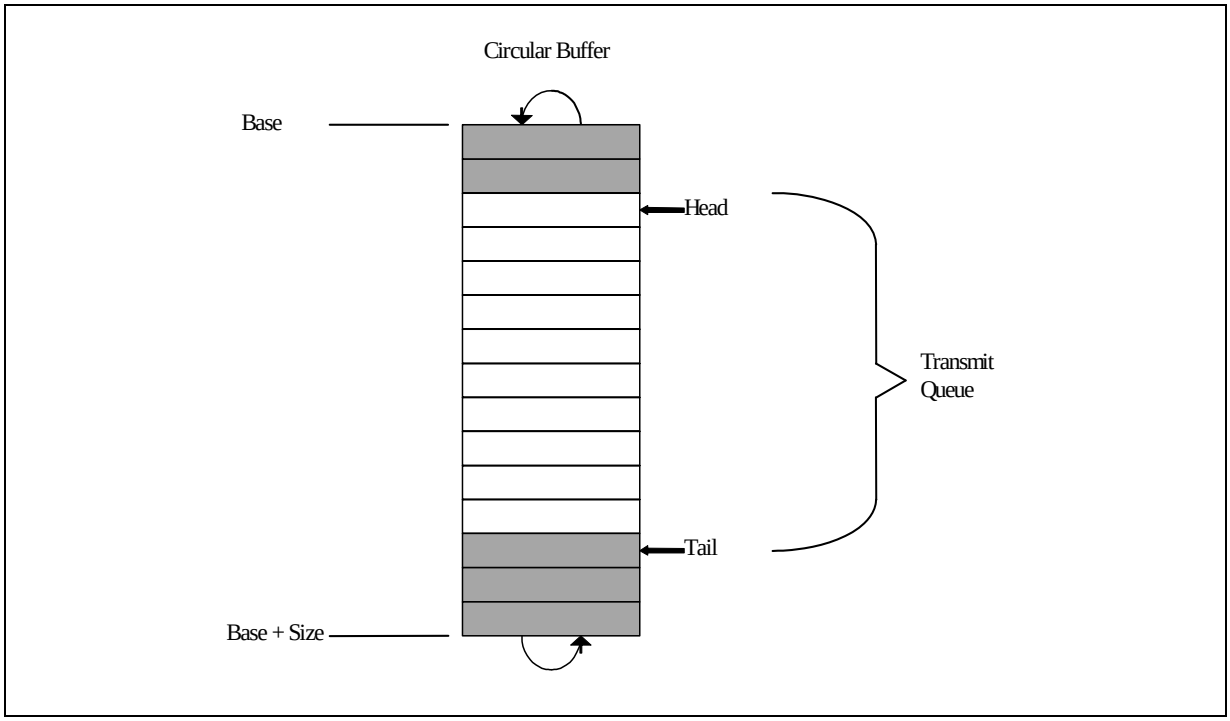
37.5.6.6 Transmit Descriptor Structure

A pair of hardware registers maintains the transmit descriptor ring in the host memory. New descriptors are added to the queue by software by writing descriptors into the circular buffer memory region and moving the tail pointer associated with that queue. The tail pointer points one entry beyond the last hardware owned descriptor. Transmission continues up to the descriptor where head equals tail at which point the queue is empty.

Descriptors passed to hardware should not be manipulated by software until the head pointer has advanced past them.

The transmit descriptor ring structure is shown in Figure 37-29. Shaded boxes represent descriptors that have been transmitted but not yet reclaimed by software. Reclaiming involves freeing up buffers associated with the descriptors.

Figure 37-29.Transmit Descriptor Ring Structure



The transmit descriptor ring is described by the following registers:

The Transmit Descriptor Base Address High Register and the Transmit Descriptor Base Address Low Register (TDBAH and TDBAL) - These registers indicate the start of the descriptor ring buffer in the host memory; this 64-bit address is aligned on a 16B boundary. Hardware ignores the lower 4 bits.

Transmit Descriptor Length Register (TDLEN) - This register determines the number of bytes allocated to the circular buffer. This value must be 0 modulo 128.

Transmit Descriptor Head Register (TDH) - This register holds a value which is an offset from the base, and indicates the in-progress descriptor. There can be up to 64K descriptors in the circular buffer. Reading this register returns the value of “head” corresponding to descriptors already loaded in the output FIFO.

Transmit Descriptor Tail Register (TDT) - This register holds a value which is an offset from the base, and indicates the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

The base register indicates the start of the circular descriptor queue and the length register indicates the maximum size of the descriptor ring. The lower seven bits of length are hard-wired to 0. Byte addresses within the descriptor buffer are computed as follows: address = base + (pointer * 16), where pointer is the value in the hardware head or tail register.

The size chosen for the head and tail registers permit a maximum of 64K descriptors, or approximately 16K packets for the transmit queue given an average of four descriptors per packet.

Once activated, hardware fetches the descriptor indicated by the hardware head register. The hardware tail register points one beyond the last valid descriptor. Software reads the head register to determine which packets (those logically before the head) have been transmitted.

Note: Software can determine if a packet has been sent by either of two methods: setting the RS bit in the transmit descriptor command field or by performing a host CPU read of the transmit head register. Checking the transmit descriptor DD bit in memory eliminates a potential race condition. All descriptor data is written to the host bus prior to incrementing the head register, but a read of the head register could “pass” the data write in systems performing write buffering. Updates to transmit descriptors use the same write path and follow all data writes. Consequently, they are not subject to the race.

Hardware pre-fetches the entire packet data prior to transmission, and updates the value of the head pointer after storing each descriptor's data in the transmit FIFO.

The process of checking for completed packets consists of one of the following:

- Scan memory.
- Read the hardware head register. All packets up to but ***excluding*** the one pointed to by head have been sent or buffered and can be reclaimed.
- Take an interrupt. An interrupt condition is generated whenever a transmit queue goes empty (ICR.TXQE). This interrupt can either be enabled or masked.

37.5.6.7 Transmit Descriptor Fetching

The descriptor processing strategy for transmit descriptors is essentially the same as for receive descriptors except that a different set of thresholds are used. Just as with receives, the number of transmit descriptors held in hardware has been increased (from 8 to 64), and the fetch and write-back algorithms modified.

When the hardware descriptor buffer is empty, a fetch will happen as soon as any descriptors are made available (host writes to the tail pointer). When the hardware descriptor buffer is nearly empty (TXDCTL.PTHRESH), a prefetch will be performed whenever enough valid descriptors (TXDCTL.HTHRESH) are available in host memory and no other DMA activity of greater priority is pending (descriptor fetches, descriptor write-backs, or packet data transfers).

When the number of descriptors in host memory is greater than the available hardware descriptor storage, the GbE may elect to perform a fetch which is not a multiple of cache line size. The hardware performs this non-aligned fetch if doing so will result in the next descriptor fetch being aligned on a cache line boundary. This allows the descriptor fetch mechanism to be most efficient in the cases where it has fallen behind software.

Note: The GbE NEVER fetches descriptors beyond the descriptor TAIL pointer. This is different from the 82542 design.

37.5.6.8 Transmit Descriptor Write-back

The descriptor write-back policy for transmit descriptors is similar to that for receive descriptors with a few additional factors. First, since transmit descriptor write-backs are optional (controlled by RS in the transmit descriptor), only descriptors which have one (or both) of these bits set will start the accumulation of write-back descriptors. Secondly, to preserve backward compatibility with the 82542, if the TXDCTL.WTHRESH value is 0, the device will write back a single byte of the descriptor (TDESCR.STA) and all other bytes of the descriptor will be left unchanged.

The benefit of delaying and then bursting transmit descriptor write-backs is small at best. In this case, it is recommended the threshold be left at the default value (0) to force immediate write-back of transmit descriptors and to preserve backward compatibility.

Descriptors are written back in one of three cases:

- TXDCTL.WTHRESH = 0 and a descriptor which has RS set is ready to be written back,
- Transmit Interrupt Delay timer expires, or
- TXDCTL.WTHRESH > 0 and TXDCTL.WTHRESH descriptors have accumulated.

For the first condition, write-backs are immediate. This is the default operation and is backward compatible with the 82542 implementation. For this case, the Transmit Interrupt delay function works as described in "Delayed Transmit Interrupts" on page 1378.

The other two conditions are only valid if descriptor bursting is enabled "TXDCTL – Transmit Descriptor Control Register" on page 1500. In the second condition, the "TIDV – Transmit Interrupt Delay Value Register" on page 1499 is used to force timely write-back of descriptors. The first packet after timer initialization starts the timer. Timer expiration flushes any accumulated descriptors and sets an interrupt event (ICR.TXDW).

For the final condition, if TXDCTL.WTHRESH descriptors are ready for write-back, the write-back is performed.

37.5.6.9 Transmit Interrupts

Hardware supplies three transmit interrupts. These interrupts are initiated via the following conditions:

Transmit Descriptor Ring empty (ICR.TXQE) - All descriptors have been processed. The head pointer is equal to the tail pointer.

Descriptor done (Transmit Descriptor Write-back (ICR.TXDW)) - Set when hardware writes back a descriptor with RS set. This is only expected to be used in cases where, for example, the streams interface has run out of descriptors and wants to be interrupted whenever progress is made.

Transmit Delayed Interrupt (ICR.TXDW) - In conjunction with IDE (Interrupt Delay Enable), the ICR.TXDW indication is delayed per the TIDV and/or TADV registers. The interrupt is set when one of the transmit interrupt countdown timers expires. A Transmit Delayed Interrupt is scheduled for a transmit descriptor with its RS and IDE bits both set. When a Transmit Delayed Interrupt occurs, the ICR.TXDW interrupt cause bit is set (just as when a Transmit Descriptor Write-back interrupt occurs). This interrupt may be masked in the same manner as the ICR.TXDW interrupt. This interrupt will be used frequently by software that performs dynamic transmit chaining, by adding packets one at a time to the transmit chain.

Note: The transmit delay interrupt is indicated with the same interrupt bit as the transmit write-back interrupt, ICR.TXDW. The transmit delay interrupt is only delayed in time as discussed above.

Transmit Descriptor Ring Low Threshold Hit (ICR.TXD_LOW) - Set when the total number of transmit descriptors available (as measured by the difference between the TX descriptor ring Head and Tail pointer) hits the low threshold specified in the TXDCTL.LWTHRESH field in the transmit descriptor control register.

37.5.6.9.1 Delayed Transmit Interrupts

This mechanism allows software the flexibility of delaying transmit interrupts until no more descriptors are added to a transmit chain for a certain amount of time, rather than when the device's head pointer catches the tail pointer. This will occur if the device is processing packets slightly faster than the software, a likely scenario for gigabit operations.

This feature is desirable, because a software driver usually has no knowledge of when it is going to be asked to send another frame, and for performance reasons, it is best to generate only one transmit interrupt after a burst of packets have been sent.

Refer to "Transmit Descriptor Formats" on page 1365 for specific details.

37.5.6.10 Transmit Checksum Off loading

The previous section on TCP Segmentation off load describes the IP/TCP/UDP checksum off loading mechanism used in conjunction with TCP Segmentation. The same underlying mechanism can also be applied as a standalone feature. The main difference in normal packet mode (non-TCP Segmentation) is that only the checksum fields in the IP/TCP/UDP headers need to be updated.

Before taking advantage of the GbE enhanced checksum off load capability, a checksum context must be initialized. For the normal transmit checksum off load feature this is performed by providing the device with a TCP/IP Context Descriptor with TSE=0. Setting TSE=0 indicates that the normal checksum context is being set, as opposed to the segmentation context. For additional details on contexts, refer to "Overview of GbE hardware "Contexts"" on page 1369.

Note: Enabling the checksum off loading capability without first initializing the appropriate checksum context will lead to unpredictable results.

Once the checksum context has been set, that context will be used for all normal packet transmissions until a new context is loaded. Also, since checksum insertion is controlled a per packet basis, there is no need to clear/reset the context.

The GbE is capable of performing two transmit checksum calculations. Typically these would be used for TCP/IP and UDP/IP packet types; however, the mechanism is general enough to support other checksums as well. Each checksum operates independently and provides identical functionality. Only the IP checksum case is discussed below.

Three fields in the TCP/IP Context Descriptor set the context of the IPv4 checksum off loading feature: IPCSS, IPCSO, and IPCSE.

IPCSS specifies the byte offset from the start of the data fetched from host memory to the first byte to be included in the checksum. Setting this value to "0" means the first byte of the data would be included in the checksum. Note that the maximum value for this field is 255. This is adequate for typical applications.

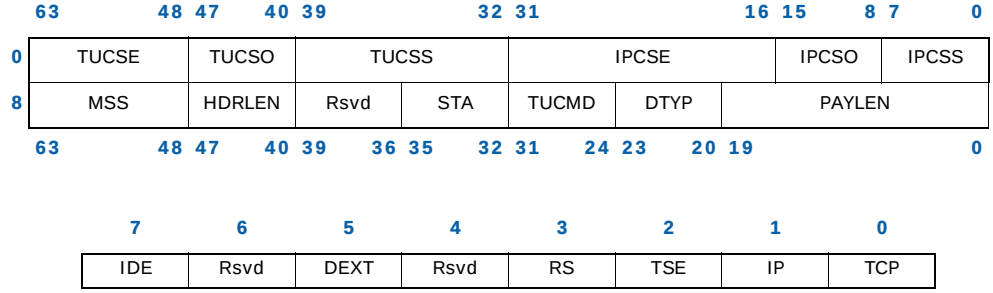
Note: The CSS value needs to be less than the total DMA length for a packet. If this is not the case, the results will be unpredictable.

The IPCSO field specifies where the resulting checksum should be placed. Again, this is limited to the first 256 bytes of the packet and must be less than or equal to the total length of a given packet. If this is not the case, the checksum will not be inserted.

IPCSE specifies where the checksum should stop. A 16-bit value supports checksum off loading of packets as large as 64KB. Setting the IPCSE field to all zeros means End-of-Packet. In this way, the length of the packet does not need to be calculated.

As mentioned in "Overview of GbE hardware "Contexts"" on page 1369, it is not necessary to set a new context for each new packet. In many cases, the same checksum context can be used for a majority of the packet stream. In this case, some performance can be gained by only changing the context on an as needed basis or electing to use the off load feature only for a particular traffic type, thereby avoiding all context descriptors except for the initial one.

Figure 37-30.Transmit Descriptor and TUCMD Field (TDESC) Layouts - (Type = 0000)



IDE: Interrupt Delay Enable

Rsvd: Reserved

DEXT: Descriptor Extension (Must be 1)

Rsvd: Reserved

RS: Report Status

TSE: TCP Segmentation Enable

IP: IPv4 Header Update Enable

TCP: Packet Type (TCP=1,UDP=0)

37.5.7 TCP Segmentation

Hardware TCP Segmentation is one of the off-loading options of the Windows 2000* TCP/IP stack. This is often referred to as “Large send” off loading. This feature allows the TCP/IP stack to pass to the network device driver a message to be transmitted that is bigger than the Maximum Transmission Unit (MTU) of the medium. It is then the responsibility of the device driver and hardware to carve the TCP message into MTU size frames that have appropriate layer 2 (Ethernet), 3 (IP), and 4 (TCP) headers. These headers must include sequence number, checksum fields, options and flag values as required. Note that some of these values (such as the checksum values) will be unique for each packet of the TCP message, and other fields such as the source IP address will be constant for all packets associated with the TCP message.

The off loading of these mechanisms to the device driver and the EP80579’s GbE will save significant CPU cycles. The device driver and the EP80579’s GbE will share the additional tasks in order to support this feature.

Prior to Windows 2000*, the Microsoft* TCP/IP stack (TCPIP.SYS) had always transferred (to the miniport device driver and eventually to the MAC) a frame with bytes up to the size of MTU for the media (up to 1514 bytes for 802.3 Ethernet packets).

Note: Although the GbE TCP segmentation off load implementation was specifically designed to take advantage of the new “TCP Segmentation offload” feature from Microsoft, the hardware implementation was made generic enough so that it could also be used to “segment” traffic from other protocols. For instance this feature could be used any time it is desirable for HW to segment a large block of data for transmission into multiple packets that contain the same generic header.

37.5.7.1 Assumptions

The GbE TCP Segmentation implementation assumes the RS bit operation is not changed. Interrupts are set after data in buffers pointed to by individual descriptors is transferred from host memory to GbE hardware.

37.5.7.2 Transmission Process

The transmission process for regular (non-TCP Segmentation packets) involves the following:

- The protocol stack receives a block of data that is to be transmitted from an application.
- The protocol stack calculates the number of packets required to transmit this block based on the MTU size of the media and required packet headers.
- For each packet of the data block:
  - The stack prepares Ethernet, IP and TCP/UDP headers.
  - The stack interfaces with the device driver and commands the driver to send the individual packet.
  - The driver gets the frame and interfaces with the hardware.
  - The hardware reads the packet from host memory (via DMA transfers).
  - The driver returns ownership of the packet to the NOS when the hardware has completed the DMA transfer of the frame (indicated by an interrupt).

The transmission process for the GbE TCP segmentation off load implementation involves the following:

- The protocol stack receives a block of data that is to be transmitted from an application.
- The stack interfaces to the device driver and passes the block down with the appropriate header information.
- The device driver sets up the interface to the hardware (via descriptors) for the TCP Segmentation context.
- The hardware transfers the packet data from host memory and performs the Ethernet packet segmentation and transmission based on offset and payload length parameters in the TCP/IP context descriptor including:
  - Packet encapsulation
  - Header generation and field updates including IP and TCP/UDP checksum generation
- The driver returns ownership of the block of data to the NOS when the hardware has completed the DMA transfer of the entire data block (indicated by an interrupt).

37.5.7.2.1 TCP Segmentation Data Fetch Control

To perform TCP Segmentation in the GbE, the DMA must ensure that the entire payload of the ***segmented*** packet will fit into the available space in the GbE Packet Buffer. The segmentation process should be performed without interruption. The DMA will do various comparisons between the payload and the Packet Buffer to ensure no interruptions occur as well as general efficiencies in the operation of the TCP Segmentation feature. The "TSPMT – TCP Segmentation Pad and Minimum Threshold Register" on page 1503 is used to allow software to program the minimum threshold required for a TCP Segmentation payload. Consideration should be made for the MTU

value when writing this field. The TSPMT register is also used to program the threshold padding overhead. This padding is necessary due to the indeterminate nature of the MTU and the associated headers.

37.5.7.3 TCP Segmentation Performance

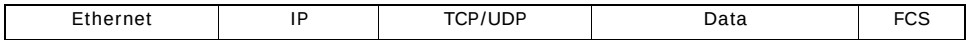
Performance improvements for a hardware implementation of TCP Segmentation off load include:

- The stack does not need to partition the block to fit the MTU size, saving CPU cycles.
- The stack only computes one Ethernet, IP, and TCP header per segment, saving CPU cycles.
- The Stack interfaces with the device driver only once per block transfer, instead of once per frame.
- Larger internal bus bursts are used which improves bus efficiency (i.e. lowering transaction overhead).
- Interrupts are easily reduced to one per TCP message instead of one per packet.
- Fewer I/O accesses are required to command the hardware.

37.5.7.4 Packet Format

Typical TCP/IP transmit window size is 8760 bytes (about 6 full size frames). Today the average size on corporate Intranets is 12-14KB, and normally the maximum window size allowed is 64KB. A TCP message can be as large as 64KB and is generally fragmented across multiple pages in host memory. The GbE partitions the data packet into standard Ethernet frames prior to transmission. The GbE also supports calculating the Ethernet, IP, TCP, and even UDP headers, including checksum, on a frame by frame basis.

Figure 37-31.TCP/IP Packet Format



Frame formats supported by the GbE include:

- Ethernet 802.3,
- IEEE 802.1q VLAN (Ethernet 802.3ac),
- Ethernet Type 2,
- Ethernet SNAP,
- IPv4 headers with options,
- IPv6 headers with IP option next headers,
- IPv6 packet tunneled in IPv4,
- TCP with options, and
- UDP with options.

VLAN tag insertion is also handled by hardware.

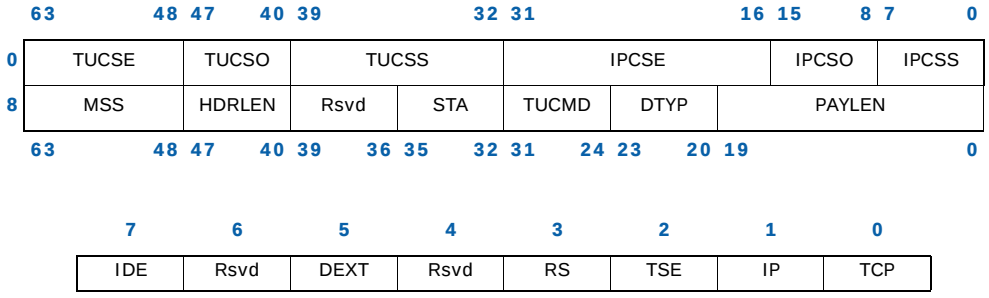
Note: UDP (unlike TCP) is not a “reliable protocol”, and fragmentation is not supported at the UDP level. UDP messages that are larger than the MTU size of the given network medium are normally fragmented at the IP layer. This is different from TCP, where large TCP messages can be fragmented at either the IP or TCP layers depending on the software implementation. The GbE has the ability to segment UDP traffic (in addition to TCP traffic), but because UDP packets are generally fragmented at the IP layer the GbE “TCP Segmentation” feature will normally not be conducive to handling UDP traffic.

Note: IP tunneled packets are not supported for Large Send operation.

37.5.7.5 TCP Segmentation Indication

Software indicates a TCP Segmentation transmission context to the hardware by setting up a TCP/IP Context Transmit Descriptor (refer to "TCP/IP Context Transmit Descriptor Format" on page 1369). The purpose of this descriptor is to provide information to the hardware to be used during the TCP segmentation off load process. The layout of this descriptor is reproduced in Figure 37-32.

Figure 37-32.TCP/IP Context Transmit Descriptor & Command Layout



Setting the TSE bit in the Command field to '1' indicates that this descriptor refers to the TCP Segmentation context (as opposed to the normal checksum off loading context). This will cause the checksum off loading, packet length, header length, and maximum segment size parameters to be loaded from the descriptor into the device.

The TCP Segmentation prototype header is taken from the packet data itself. Software must identity the type of packet that is being sent (IP/TCP, IP/UDP, other), calculate appropriate checksum off loading values for the desired checksums, and calculate the length of the header which is prepended. The header may be up to 240 bytes in length.

37.5.7.6 TCP Segmentation Data Descriptors

37.5.7.6.1 TCP Segmentation Source Data

Once the TCP Segmentation context has been set, the next descriptor provides the initial data to transfer. This first descriptor(s) must point to a packet of the type indicated. Furthermore, the data it points to may need to be modified by software as it will serve as the prototype header for all packets within the TCP Segmentation context. The following sections describe the supported packet types and the various updates which are performed by hardware. This should be used as a guide to determine what must be modified in the original packet header to make it a suitable prototype header.

The following summarizes the fields considered by the driver for modification in constructing the prototype header.

- IPv4 Header
  - — Length should be set to zero
  - — Identification Field should be set as appropriate for first packet of send (if not already)
  - — Header Checksum should be zeroed out unless some adjustment is needed by the driver
- IPv6 Header
  - — Length should be set to zero
- TCP Header

— Sequence Number should be set as appropriate for first packet of send (if not already)

— PSH, and FIN flags should be set as appropriate for LAST packet of send

— TCP Checksum should be set to the partial pseudo-header checksum as follows (there is a more detailed discussion of this in "IP and TCP/UDP Headers" just before Figure 37-40:

Figure 37-33.TCP Partial Pseudo-Header Checksum for IPv4

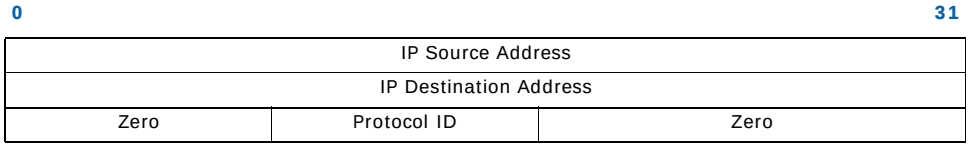
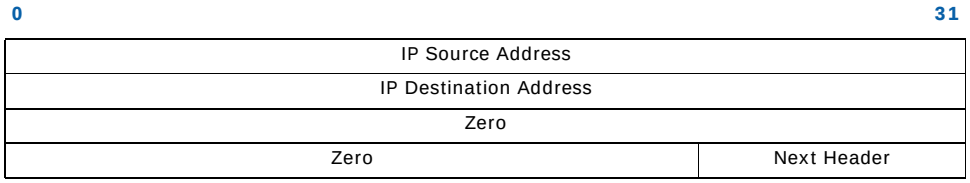


Figure 37-34.TCP Partial Pseudo-Header Checksum for IPv6



- UDP Header
  — Checksum should be set as in TCP header, shown in Figure 37-33 and Figure 37-34.

The GbE DMA function will fetch the Ethernet, IP, and TCP/UDP prototype header information from the initial descriptor(s) and save them in hardware for individual packet header generation. The following sections describe the updating process performed by the hardware for each frame sent using the TCP Segmentation capability.

37.5.7.6.2 TCP Segmentation Use of Multiple Data Descriptors

TCP Segmentation allows the packet to be segmented to be describe more than one data descriptor. A large packet contained in a single virtual-address buffer may be more simply described by a series of data descriptors, each referencing a single physical address page.

There is only one requirement for multiple data descriptors for TCP segmentation:

- If multiple data descriptors are used to describe the IP/TCP/UDP header section, each descriptor must describe one or more complete headers; descriptors referencing only parts of headers are not supported.

Note: It is recommended that the entire header section, as described by the TCP Context Descriptor HDRLEN field, be coalesced into a single buffer and described using a single data descriptor.

37.5.7.7 IP and TCP/UDP Headers

This section outlines the format and content for the IP, TCP and UDP headers. The GbE requires baseline information from the device driver in order to construct the appropriate header information during the segmentation process.

Header fields that are modified by the GbE are highlighted in the figures below.

Note: The IPv4 header is first shown in the traditional (i.e., RFC 791) representation, and because byte and bit ordering is confusing in that representation. The IP header is also shown in Little-Endian format, since the actual data will be fetched from memory in Little-Endian format.

Figure 37-35.IPv4 Header (Traditional Representation)

0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	20 21 22 23	24 25 26 27	28 29 30 31
Version	IP Hdr Length	TYPE of Service	Total Length				
Identification				Flags	Fragment Offset		
Time to Live		Layer 4 Protocol ID		Header Checksum			
Source Address							
Destination Address							
Options							

Figure 37-36.IPv4 Header (Little-Endian Order)

Byte 3								Byte 2								Byte 1								Byte 0							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
LSB								Total Length							MSB	TYPE of Service								Version				IP Hdr Length			
Fragment Offset Low								RSV	NF	MF	Fragment Offset High					LSB								Identification							MSB
Header Checksum																Layer 4 Protocol ID								Time to Live							
Source Address																															
Destination Address																															
Options																															

Figure 37-35 and Figure 37-36: Flag Bit Definition:

- MF: More Fragments (hardware does not evaluate or change this bit)
- NF: No Fragments (hardware does not evaluate or change this bit)
- RSV: Reserved

Note: The IPv6 header is shown in the traditional (i.e. RFC 2460), Big-Endian representation. The actual data will be fetched from memory in Little-Endian format, similar to the IPv4 header shown above.

Figure 37-37.IPv6 Header (Traditional Representation)

0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	20 21 22 23	24 25 26 27	28 29 30 31
Version	Traffic Class	Flow Label					
Payload Length				Next Header		Hop Limit	
Source Address							
Destination Address							

A TCP or UDP frame uses a 16 bit wide one's complement checksum. The checksum word is computed on the outgoing TCP or UDP header and payload, and on the Pseudo Header. Refer to "Transmit Checksum Off loading with TCP Segmentation" on page 1388 for details on checksum computations.

Note: TCP requires the use of checksum, where it is optional for UDP.

Note: The TCP header is first shown in the traditional (i.e. RFC 793) representation, and because byte and bit ordering is confusing in that representation, the TCP header is also shown in Little-Endian format. The actual data will be fetched from memory in Little-Endian format.

Figure 37-38.TCP Header (Traditional Representation)

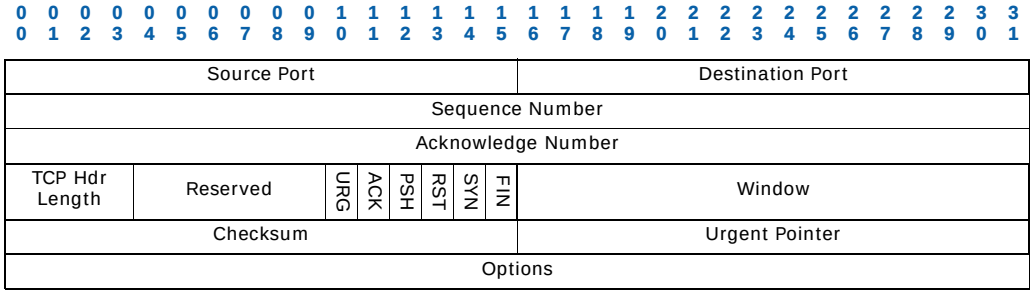
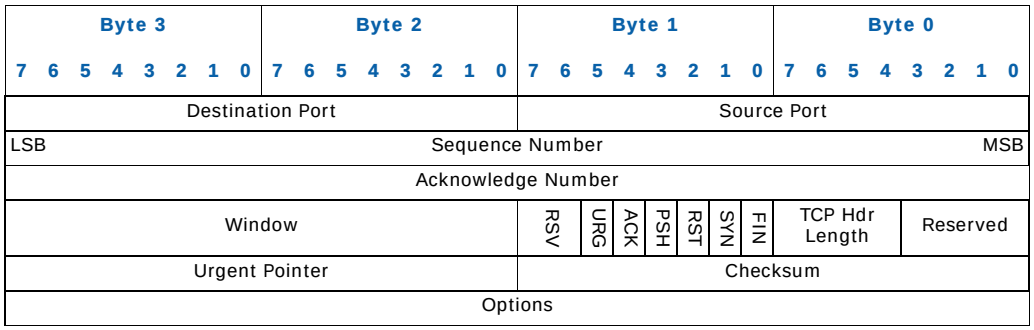


Figure 37-39.TCP Header (Little-Endian Order)



The TCP header is always a multiple of 32 bit words. TCP options may occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum.

The checksum also covers a pseudo header conceptually prefixed to the TCP Header (see Figure 37-40 and Figure 37-41 below). The IPv4 pseudo header contains the IPv4 Source Address, the IPv4 Destination Address, the IPv4 Protocol field, and TCP Length. The IPv6 pseudo header contains the IPv6 Source Address, the IPv6 Destination Address, the IPv6 Payload Length, and the IPv6 Next Header field. Software pre-calculates the partial pseudo header sum, which includes IP SA, DA and protocol type/ next header, but NOT the TCP/Payload length, and stores this value into the TCP checksum field of the packet.

Note: When calculating the TCP pseudo header, the byte ordering can be tricky. One common question is whether the Protocol ID field is added to the "lower" or "upper" byte of the 16 bit sum. The Protocol ID field should be added the least significant byte (LSB) of the 16 bit pseudo header sum, where the most significant byte (MSB) of the 16 bit sum is the byte that corresponds to the first checksum byte out on the wire.

The TCP Length field is the TCP Header Length including option fields plus the data length in bytes, which is calculated by hardware on a frame by frame basis. The TCP Length does not count the 12 bytes of the pseudo header. The TCP length of the packet is determined by hardware as:

TCP Length = Payload + HDRLEN - TUCSS

"Payload" is normally MSS except for the last packet where it represents the remainder of the payload.

Figure 37-40.TCP Pseudo Header Content (Traditional Representation)

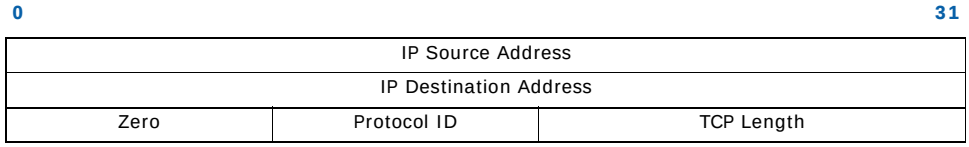
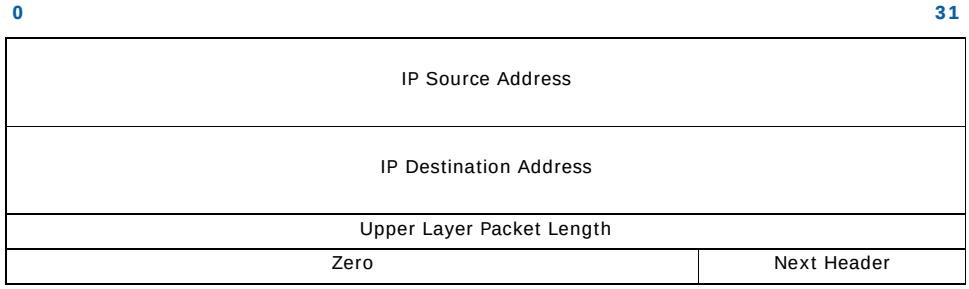


Figure 37-41.TCP Pseudo-Header Content for IPv6



Note: The IP Destination Address is the final destination of the packet. Therefore, if a routing header is used, the last address in the route list is used in this calculation. The upper-layer packet length is the length of the TCP header and TCP payload.

The UDP header is always 8 bytes in size with no options.

Figure 37-42.UDP Header (Traditional Representation)

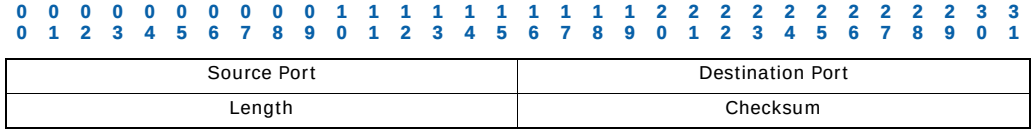
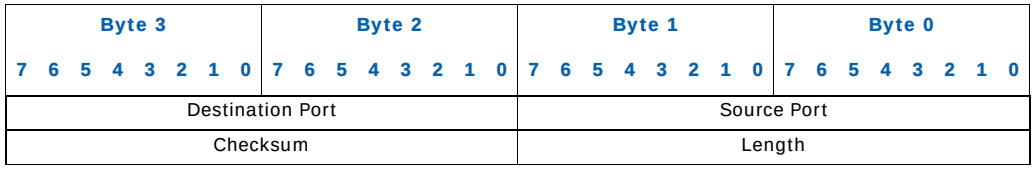


Figure 37-43.UDP Header (Little-Endian Order)



UDP pseudo header has the same format as the TCP pseudo header. The IPv4 pseudo header conceptually prefixed to the UDP header contains the IPv4 source address, the IPv4 destination address, the IPv4 protocol field, and the UDP length (same as the TCP Length discussed above). The IPv6 pseudo header for UDP is the same as the IPv6 pseudo header for TCP. The checksum procedure is the same as is used in TCP.

Figure 37-44.UDP Pseudo Header Diagram for IPv4

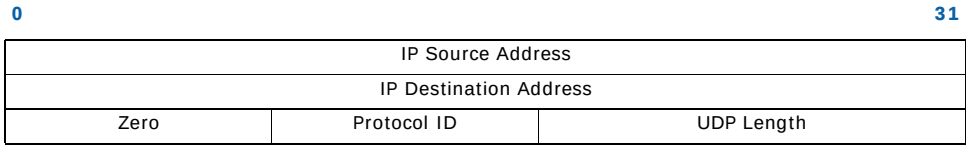
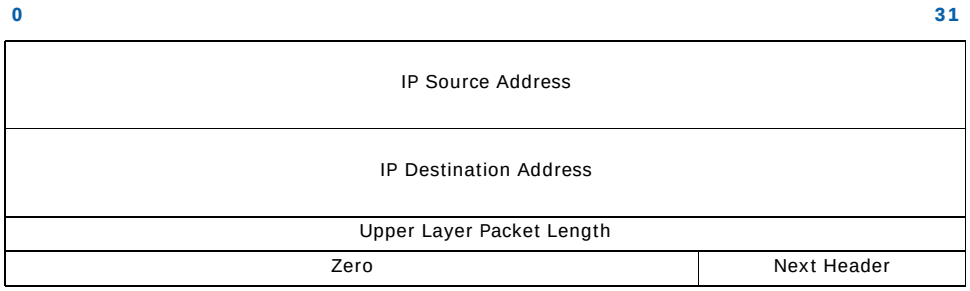


Figure 37-45.UDP Pseudo-Header Diagram for IPv6



Note: The IP Destination Address is the final destination of the packet. Therefore, if a routing header is used, the last address in the route list is used in this calculation. The upper-layer packet length is the length of the UDP header and UDP payload.

Unlike the TCP checksum, the UDP checksum is optional. Software must set the TXSM bit in the TCP/IP Context Transmit Descriptor to indicate that a UDP checksum should be inserted. Hardware will not update the UDP checksum unless the TXSM bit is set.

37.5.7.8 Transmit Checksum Off loading with TCP Segmentation

The GbE supports checksum off-loading as a component of the TCP Segmentation off load feature and as a standalone capability. Refer to "TCP/IP Context Transmit Descriptor Format" on page 1369 for details on the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP Segmentation.

The GbE supports IP and TCP/UDP header options in the checksum computation for packets that are derived from the TCP Segmentation feature.

Note: The GbE is capable of computing one level of IP header checksum and one TCP/UDP header and payload checksum. In the case of multiple IP headers, the driver will have to compute all but one IP header checksum. The GbE calculates checksums on the fly on a frame by frame basis and inserts the result in the IP/TCP/UDP headers of each frame. TCP and UDP checksum are a result of performing the checksum on all bytes of the payload and the pseudo header.

Three specific types of checksum are supported by the hardware in the context of the TCP Segmentation off load feature IPv4 checksum (IPv6 does not have a checksum), TCP checksum, and UDP checksum.

Each packet that is sent via the TCP segmentation off load feature optionally includes the IPv4 checksum and either the TCP or UDP checksum.

All checksum calculations use a 16-bit wide one's complement checksum calculated of all 16-bit words in the range of CSS to CSE, including the checksum field itself. The checksum field is written with this hardware computed value.

37.5.7.9 IP/TCP/UDP Header Updating

IP/TCP/UDP header is updated for each outgoing frame based on the IP/TCP header prototype which hardware DMA's from the first descriptor(s) and stores in hardware. The IP/TCP/UDP headers are fetched from host memory into a 240 byte header buffer once for each TCP segmentation context (for performance reasons, this header is not fetched again for each additional packet that will be derived from the TCP segmentation process). The checksum fields and other header information are later updated on a frame by frame basis. The updating process is performed concurrently with the packet data fetch.

The following sections define what fields are modified by hardware during the TCP Segmentation process by the GbE.

37.5.7.9.1 TCP/IP/UDP Header for the First Frame

The hardware makes the following changes to the headers of the first packet that is derived from each TCP segmentation context.

- IPv4 Header
  - — IP Total Length = MSS + HDRLEN - IPCSS
  - — IP Checksum
- IPv6 Header
  - — Payload Length = MSS + HDRLEN - IPCSS
- TCP Header
  - — Sequence Number: The value is the Sequence Number of the first TCP byte in this frame.
  - — If FIN flag = 1, it is cleared in the first frame.
  - — If PSH flag =1, it is cleared in the first frame.
  - — TCP Checksum
- UDP Header
  - — UDP length: MSS + HDRLEN - TUCSS
  - — UDP Checksum

37.5.7.9.2 TCP/IP/UDP Header for the Subsequent Frames

The hardware makes the following changes to the headers for subsequent packets that are derived as part of a TCP segmentation context:

Note: Number of bytes left for transmission = PAYLEN - (N * MSS). Where N is the number of frames that have been transmitted.

- IPv4 Header
  - — IP Identification: incremented from last value (wraps around)
  - — IP Total Length = MSS + HDRLEN - IPCSS
  - — IP Checksum
- IPv6 Header
  - — Payload Length = MSS + HDRLEN - IPCSS
- TCP Header
  - — Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.

  - If FIN flag = 1, it is cleared in these frames.
  - If PSH flag =1, it is cleared in these frames.
  - TCP Checksum
- UDP Header
  - UDP Length: MSS + HDRLEN - TUCSS
  - UDP Checksum

37.5.7.9.3 TCP/IP/UDP Header for the Last Frame

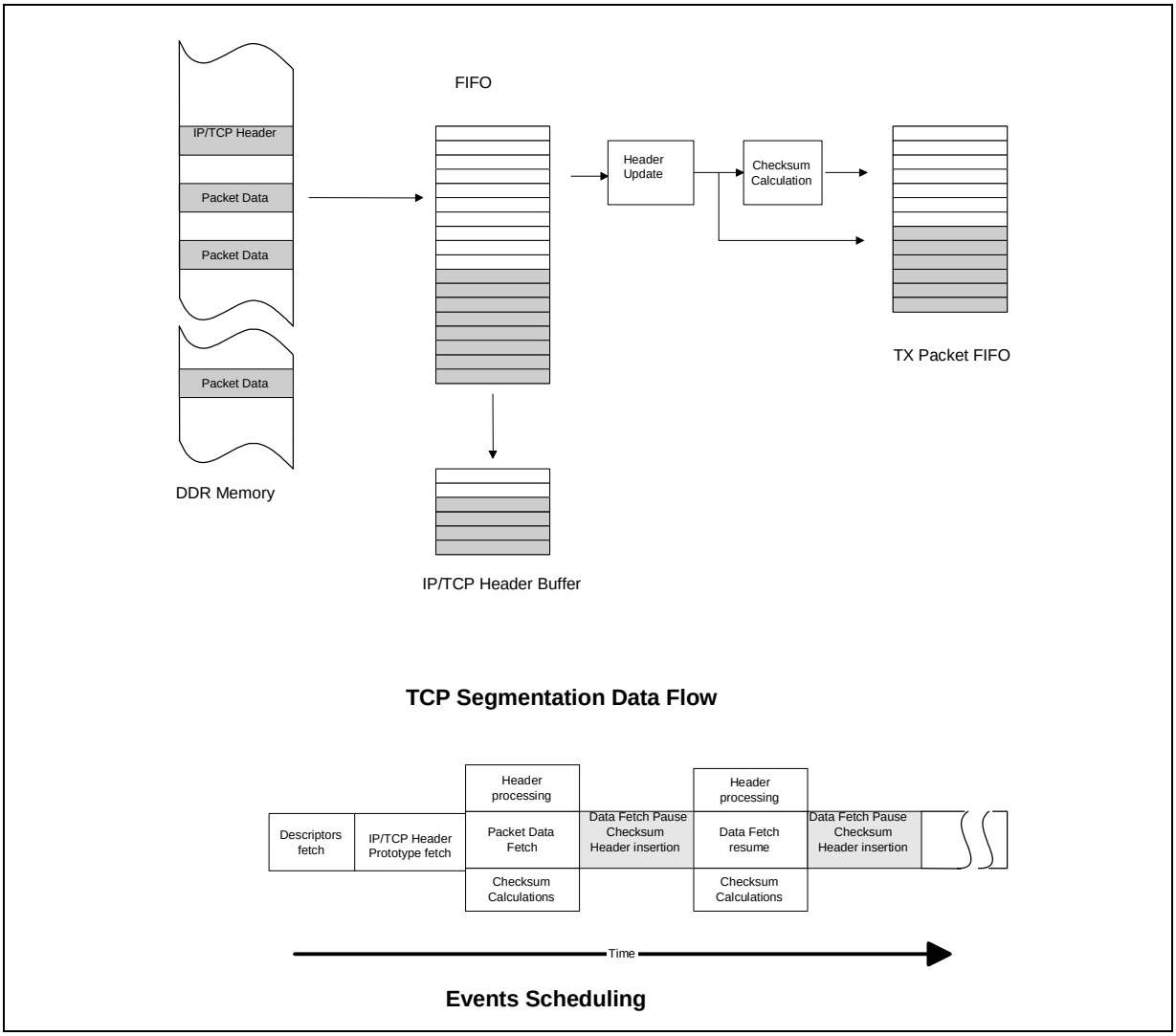
The hardware makes the following changes to the headers for the last frame of a TCP segmentation context:

Note: Last frame payload bytes = PAYLEN - (N * MSS)

- IPv4 Header
  - IP Total Length = (last frame payload bytes + HDRLEN) - IPCSS
  - IP Identification: incremented from last value (wraps around)
  - IP Checksum
- IPv6 Header
  - Payload Length = MSS + HDRLEN - IPCSS
- TCP Header
  - Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
  - If FIN flag = 1, set it in this last frame
  - If PSH flag =1, set it in this last frame
  - TCP Checksum
- UDP Header
  - UDP length: (last frame payload bytes + HDRLEN) - TUCSS
  - UDP Checksum

37.5.7.10 Data Flow

Figure 37-46.Data Flow



37.5.8 Ethernet Interfaces

The GbE MAC provides a complete CSMA/CD function supporting IEEE 802.3 (10Mbps), 802.3u (100Mbps), 802.3z and 802.3ab (1000Mbps) implementations. The device performs all of the functions required for transmission, reception and collision handling called out in the standards.

The GbE may be configured to be used with different media interfaces. The following native configurations are supported:

- External RGMII/RMII device (these interfaces are provided via gasket between the GbE's native GMII/MII interfaces)

Selection between the various configurations is programmable via the MAC Extended Device Control Register (CTRL_EXT.LINK_MODE bits).

TGMII/MII interface used to communicate between the MAC and he RGMII/RMII gasket supports 10/100/1000 Mbps operation, with both half- and full-duplex operation at 10/100 Mbps, and full-duplex operation at 1000 Mbps.

Note: The GbE MAC is optimized for full-duplex operation in 1000 Mbps mode. Half-duplex 1000 Mbps operation is not supported.

37.5.8.1 MAC/PHY GMII/MII Interface

The GbE MAC communicates through a GMII/MII interface which may be configured for either 1000 Mbps operation (GMII) or 10/100 Mbps (MII) mode of operation. For proper network operation, both the internal MAC and the external PHY must be properly configured to identical speed & duplex settings. Additionally, the translators may need to be configured in the "CTRL_AUX – Auxiliary Device Control/Status Register". All MAC configuration is performed using device control registers mapped into system memory. The PHY will either auto-negotiate the link with the link partner's PHY at the other end of the copper line, or will be forced into it's configuration by software.

37.5.8.1.1 GMII - 1000 Mbps Operation

During 1000Mbps operation, the MAC/PHY communication occurs via an interface utilizing a pair of 8-bit buses operating at 125 MHz and accompanied by a handful of additional clocks and/or qualifiers. This signaling (GMII mode) includes the following communication:

CRS (carrier sense): Carrier sense is detected by the PHY and indicates activity on the cable, either incoming or outgoing. This signal is driven by the PHY to the MAC to enable the MAC to generate link status-change alerts, and suspend transmit / ignore receive symbols when no link is present.

COL (collision detection): Collision detection is performed by the PHY, signaled to the MAC upon detection of a collision on the medium, and remains asserted while the collision condition persists. For half-duplex operation, COL indicates detection of simultaneous transmission and reception. Since collisions do not occur between full-duplex transceivers, the GbE MAC ignores any errant collision-signaling when in full-duplex mode.

TX_ER (transmit code error): This signaling is used by the MAC to indicate carrier extension and IPG during packet bursts to the PHY, as well as to force propagation of transmit errors. Note that the GbE will not transmit error codes.

TX_EN (transmit enable): This signal is asserted from the MAC to the PHY while transmitting, and used to indicate when the MAC is presenting frame data on the GMII interface to the PHY for transmission. The MAC asserts TX_EN synchronously with the first byte of the preamble, and it remains asserted until the final data byte in a frame, deasserted after the final byte.

GTX _CLK (transmit data clock): In GMII mode, the MAC provides a 125 MHz transmit clock to the PHY accompanying any transmit data.

TX_DATA (transmit data): Data is transmitted from the MAC to PHY in 8-bit quantities at 125 MHz when in GMII mode.

RX_CLK (receive clock): In GMII mode, receive data provided from the PHY to the MAC is accompanied by a 125 MHz receive clock.

RX_DATA (receive data): Data received by the PHY is transferred to the MAC in 8-bit quantities at 125 MHz in GMII mode.

RX_ER (receive error): Receive errors are detected by the PHY and signaled to the MAC. Receive errors may include link coding errors, or any other error detected by the PHY. If receive errors are signaled during packet reception, the MAC can be configured to either receive or drop these packets.

RX_DV (receive data valid): This signal is asserted from the PHY to the MAC to transfer valid frame data to the MAC. It is asserted from the first through the final bytes of a frame, de-asserted after the final byte. The PHY asserts carrier sense with this data-valid signal de-asserted to indicate to the MAC reception of broken packet headers (fragments).

37.5.8.1.2 MII - 10/100 Mbps Operation

During 10/100 Mbps operation (MII mode), the communication between the GbE MAC and the external PHY occurs via the same interface as in GMII mode, used in a similar way. In MII mode, transmit and receive data is transferred in nibble-wide (4-bit) quantities instead of 8-bit quantities, and at either 25Mhz (100 Mbps operation) or 2.5 MHz (10 Mbps operation).

Differences in signaling between GMII and MII modes are as follows:

MTX_CLK (transmit clock): In MII mode, this clock is supplied from the PHY to the MAC, and is used by the MAC for transmit-data synchronization. This clock operates at either 25MHz (for 100Base-T) or 2.5MHz (for 10BASE-T).

TX_DATA (transmit data): Transmit data is transferred from the MAC to the PHY in 4-bit (nibble) quantities at either 25 MHz or 2.5 MHz. in MII mode.

RX_CLK (receive clock): In MII mode, the receive clock provided from the PHY to the MAC operates at either 25 MHz (for 100BASE-T) or 2.5MHz (for 10BASE-T).

RX_DATA (receive data): Receive data is transferred from the PHY to the MAC in 4-bit (nibble) quantities at either 25 MHz or 2.5 MHz in MII mode.

If a pin is listed in GMII mode, but not MII mode, then it functions identically to GMII mode.

37.5.8.2 Duplex Operation

The GbE supports half-duplex MII mode and full-duplex GMII/MII mode. Configuration of the duplex operation of the device must be programmed by software.

37.5.8.2.1 Full Duplex

All aspects of the IEEE 802.3, 802.3u, 802.3z, and 802.3ab specifications are supported in full duplex operation. During full duplex operation, the GbE may transmit and receive packets simultaneously across the link interface.

In full-duplex GMII/MII mode, transmission and reception are delineated independently by the control signals. Transmission starts upon the assertion of TX_EN which indicates there is valid data on the TX_DATA bus driven from the MAC to the PHY. Reception is signaled by the PHY by the assertion of the RX_DV signal which indicates valid receive data on the RX_DATA lines to the MAC.

The GbE can receive carrier-extended packets, although it cannot transmit carrier-extended packets. Note that errors received the extended portion of the packet (carrier-extend errors) will not be detected by the GbE.

37.5.8.2.2 Half Duplex

The GbE MAC may operate in half duplex when configured for MII mode, but GMII mode does not support half duplex operation.

In half duplex operation, the MAC attempts to avoid contention with other traffic on the link by monitoring the CRS signal provided by the PHY and deferring to passing traffic. When the CRS signal is de-asserted or after a sufficient inter-packet gap (IPG) has elapsed after a transmission, frame transmission may begin. The MAC signals the PHY with TX_EN at the start of transmission.

In the case of a collision, the PHY detects the collision and asserts the COL signal to the MAC. Transmission of the frame stops within four link clock times, and the GbE sends a JAM sequence onto the link. After the end of a collided transmission, the GbE will back off and attempt to retransmit per the standard CSMA/CD method. Note that the re-transmissions are done from the data stored internally in the GbE transmit packet buffer (no re-access to the data in host memory is necessary).

In the case of a successful transmission, the GbE is ready to transmit any other frame(s) queued in the transmit FIFO, after the minimum inter-frame spacing (IFS) of the link has elapsed.

During transmit, the PHY is expected to signal a carrier-sense (assert the CRS signal) back to the MAC before one slot time has elapsed. The transmission will complete successfully even if the PHY fails to indicate CRS within the slot time window. If this situation occurs, the PHY may either be configured incorrectly or be in a link down situation. Such an event will be counted in the statistic register space, refer to "TNCRS – Transmit with No CRS Count Register" on page 1510.

MII mode half duplex reception occurs exactly as indicated in "Full Duplex" on page 1393. Reception is signaled by the PHY by the assertion of the RX_DV signal which indicates valid receive data on the RX_DATA lines to the MAC.

37.5.8.3 Physical Layer Auto-Negotiation & Link Setup Features

The method for configuring the link between two link partners is highly dependent on the mode of operation as well as the functionality provided by the specific physical layer device. For GMII/MII mode, the PCS and Auto-Negotiation functions are expected to be maintained within the external PHY.

Configuration of the link may be accomplished by several methods ranging from software's forcing link settings, to software-controlled negotiation, to auto-negotiation initiated by the PHY. The following sections describe processes of bringing the link up including configuration of the GbE and the transceiver, as well as the various methods of determining duplex and speed configuration in order to configure the MAC.

When operating in a GMII/MII mode, the external PHY performs auto-negotiation per 802.3ab clause 40 and extensions to clause 28. Link resolution is obtained by software from the PHY after the link has been established and programs the MAC with these settings.

37.5.8.3.1 GMII/MII/Copper Link Configuration

When operating in GMII/MII mode, link configuration is generally determined by Auto-Negotiation between the PHY and it's link partner. The driver must poll the configuration in cases after a successful link is established or the user desires to manually configure the link. The following sections discuss the methods of link configuration for copper PHY operation.

37.5.8.3.2 GMII/MII/Copper Auto-Negotiation (Speed, Duplex, Flow-Control)

When using a copper PHY, the PHY performs the Auto-Negotiation function. The actual operational details of this operation are described in the IEEE P802.3ab draft standard, and are not included here.

Auto-Negotiation provides a method for two link partners to exchange information in a systematic manner in order to establish a link configuration providing the highest level of functionality supported by both partners. Once configured, the link partners exchange configuration information to resolve link settings such as:

- Speed (10/100/1000 Mbps)

- Duplex (Full or Half)
- Flow Control Operation

PHY specific information required for establishing the link is also exchanged.

Note: If flow control is enabled in the GbE, the settings for the desired flow control behavior must be set by software in the PHY registers and Auto-Negotiation restarted. After Auto-Negotiation is complete, the driver must read the PHY registers to determine the resolved flow control behavior of the link and reflect these in the MAC register settings (CTRL.TFCE and CTRL.RFCE).

Note: There are two recommended methods for coordination between the PHY/PHY Auto-Negotiation and the GbE. First, the PHY can be programmed to issue an MDINT to the host CPU when the link changes. When the EP80579 receives the MDINT indicator via an interrupt, the ISR routine will poll the PHY's MDIO interface and programs the GbE accordingly. Second, software may continuously poll the PHY's configuration registers through the MDIO interface and reprogram the GbE when the configuration changes from the previous state.

MAC Speed Resolution

For proper link operation, both the MAC and PHY must be configured for the same speed of link operation. The speed of the link may be determined and set by one of the following methods:

- Hardware uses the default MAC speed of 1 Gbps and no software action is necessary, or
- Software reads the PHY registers to determine the PHY's Auto-Negotiated speed and configures the MAC by setting CTRL.FRCSPEED to 1 and CTRL.SPEED to the proper speed value (refer to "CTRL – Device Control Register" on page 1438 for details), or
- Software writes the non Auto-Negotiated PHY registers with the desired speed and configures the MAC by to the same setting by programming CTRL.FRCSPEED to 1 and CTRL.SPEED to the proper speed value (refer to "CTRL – Device Control Register" on page 1438 for details).

Note: Forcing the MAC speed using CTRL.FRCSPD can yield non-functional links if the MAC and PHY are not operating at the same speed/configuration.

Note: The forcing of the speed settings by CTRL.SPEED may also be accomplished by setting the CTRL_EXT.SPD_BYPS bit. This bit bypasses the MAC's internal clock switching logic and allows the driver complete control of when the speed setting takes place. The CTRL.FRCSPD bit uses the MAC's internal clock switching logic, which does slightly delay the affect of the speed change.

MAC Full/Half Duplex Resolution

The duplex configuration of the link is also resolved by the PHY during the Auto-Negotiation process. For proper link operation, both the MAC and PHY must be also configured for the same duplex configuration. The duplex configuration of the link may be determined and set by one of the following methods:

- Hardware uses the default MAC speed of Full Duplex and no software action is necessary (the PHY configuration must be KNOWN to be Full-Duplex for this option), or
- Software reads the PHY registers to determine the PHY's Auto-Negotiated duplex setting and configures the MAC by setting CTRL.FRCDPLX to 1 and CTRL.FD to the proper value (refer to "CTRL – Device Control Register" on page 1438 for details), or

- Software writes the non Auto-Negotiated PHY registers with the desired duplex configuration and configures the MAC by to the same setting by programming CTRL.FRCDPLX to 1 and CTRL.FD to the proper speed value (refer to "CTRL – Device Control Register" on page 1438 for details).

Using PHY Registers

The driver may be required under some circumstances to read from, or write to, the MII management registers in the PHY. These accesses are performed via the MDIO registers in the GCU (refer to the GCU EAS Chapter for details). The MII registers allow the driver to have direct control over the PHY's operation which may include:

- Resetting the PHY,
- Setting preferred link configuration for advertisement during the Auto-Negotiation process,
- Restarting the Auto-Negotiation process,
- Reading Auto-Negotiation status from the PHY, and
- Forcing the PHY to a specific link configuration.

The set of PHY management registers required for all PHY devices may be found in the IEEE P802.3ab draft standard.

Comments Regarding Forcing Link

Forcing link in GMII/MII mode requires the driver to configure both the MAC and PHY in a consistent manner with respect to each other as well as the link partner. After initialization, the driver configures the desired modes in the MAC, then accesses the PHY registers to set the PHY to the same configuration.

Before enabling the link, the speed and duplex settings of the MAC may be forced by software using the CTRL.FRCSPD, CTRL.FRCDPX, CTRL.SPEED, and CTRL.FD bits. After the PHY and MAC have both been configured, the driver should write a 1 to the CTRL.SLU bit.

37.5.8.4 10/100Mbps Specific Performance Enhancements

37.5.8.4.1 Adaptive IFS

The GbE supports back-to-back transmit Inter-Frame-Spacing (IFS) of 960 ns in 100 Mbit operation and 9.6 us in 10M bit operation. Although back-to-back transmission is normally desirable, sometimes it can actually hurt performance in half-duplex environments due to excessive collisions. Excessive collisions are likely to occur in environments where one station is attempting to send large frames back-to-back, while another station is attempting to send acknowledge (ACK) packets.

The GbE contains the AIT register that enables the implementation of a driver based adaptive IFS algorithm for collision reduction, which is similar to Intel's other Ethernet products (e.g. PRO/100 adapters). Refer to "AIT – Adaptive IFS Throttle Register" on page 1495 for explicit details. Essentially, the Adaptive IFS throttles back-to-back transmissions in the transmit MAC and delays their transfer to the CSMA/CD transmit function, and thus can be used to delay the transmission of back-to-back packets on the wire. Normally, this register should be set to 0. However, if additional delay is desired between back-to-back transmits, then this register may be set with a value greater than zero.

The AIT.AIFS provides a similar function to TIPG.IGPT (see "TIPG – Transmit IPG Register" on page 1493). However this Adaptive IFS throttle register counts in units of GTX/MTX_CLK clocks (which are 8ns, 80ns, 800ns for 10, 100, 1000 Mb/s mode respectively), and is 16 bits wide, thus providing a greater maximum delay value.

Using values lower than a certain minimum (determined by the ratio of GTX/MTX_CLK clock to link speed), will have no effect on back-to-back transmission. This is because the device will not start transmission until the minimum IEEE IFS (9.6 us at 10Mb, 960 ns at 100 b, and 96 ns at 1 Gb) has been met regardless of the value of Adaptive IFS. For example, if the GbE is configured for 100 Mbps operation, the minimum IEEE IFS at 100 Mbps is 960 nanoseconds. Setting AIFS to a value of 10 (decimal) would not effect back-to-back transmission time on the wire, because the 800 ns delay introduced (10 * 80n s = 800 ns) is less than the minimum IEEE IFS delay of 960 ns. However, setting this register with a value of 20, which corresponds to 1600 ns for the above example, would delay back-to-back transmits because the ensuing 1600 ns delay is greater than the minimum IFS time of 960 ns.

It is important to note that this register has no effect on transmissions that occur immediately after receives, or on transmissions that are not back-to-back (unlike the TIPG.IPGR1 and TIPG.IPGR2 settings -- see "TIPG – Transmit IPG Register" on page 1493). In addition, Adaptive IFS also has no effect on re-transmission timing (re-transmissions occur after collisions). Therefore, AIT.AIFS is only enabled in back-to-back transmission.

Note: The AIT.AIFS value is not additive to the TIPG.IPGT value; instead, the actual IPG equals the larger of AIT.AIFS and TIPG.IPGT.

37.5.8.5 Flow Control

Flow control as defined in 802.3x, as well as the specific operation of asymmetrical flow control defined by 802.3z, is supported in the MAC. The following six registers are defined for the implementation of flow control:

The Flow Control Address High Register and the Flow Control Address Low Register (FCAH and FCAL) - 6 byte flow control multicast address

Flow Control Type Register (FCT) - 16 bit field to indicate flow control type

Flow Control Receive Threshold High Register (FCRTH) - 13 bit high water mark indicating receive buffer fullness

Flow Control Receive Threshold Low Register (FCRTL) - 13 bit low water mark indicating receive buffer emptiness

Flow Control Transmit Timer Value Register (FCTTV) - 16 bit timer value to include in transmitted PAUSE frame

Flow control is implemented as a means of reducing the possibility of receive buffer overflows which result in the dropping of received packets, and allows for local controlling of network congestion levels. This may be accomplished by sending an indication to a transmitting station of a nearly-full receive buffer condition at a receiving station.

The implementation of asymmetric flow control allows for one link partner to send flow control packets while being allowed to ignore their reception; i.e. not required to respond to PAUSE frames.

37.5.8.5.1 MAC Control Frames & Reception of Flow Control Packets

Three comparisons are used to determine the validity of a flow control frame:

- A match on the six byte multicast address for MAC Control Frames or to the station address of the device (Receive Address Register 0).
- A match on the type field.
- A comparison of the MAC Control Opcode field.

The 802.3x standard defines the MAC Control Frame multicast address as 01-80-C2-00-00-01. This address must be loaded into the Flow Control Address High Register and the Flow Control Address Low Register (FCAH and FCAL).

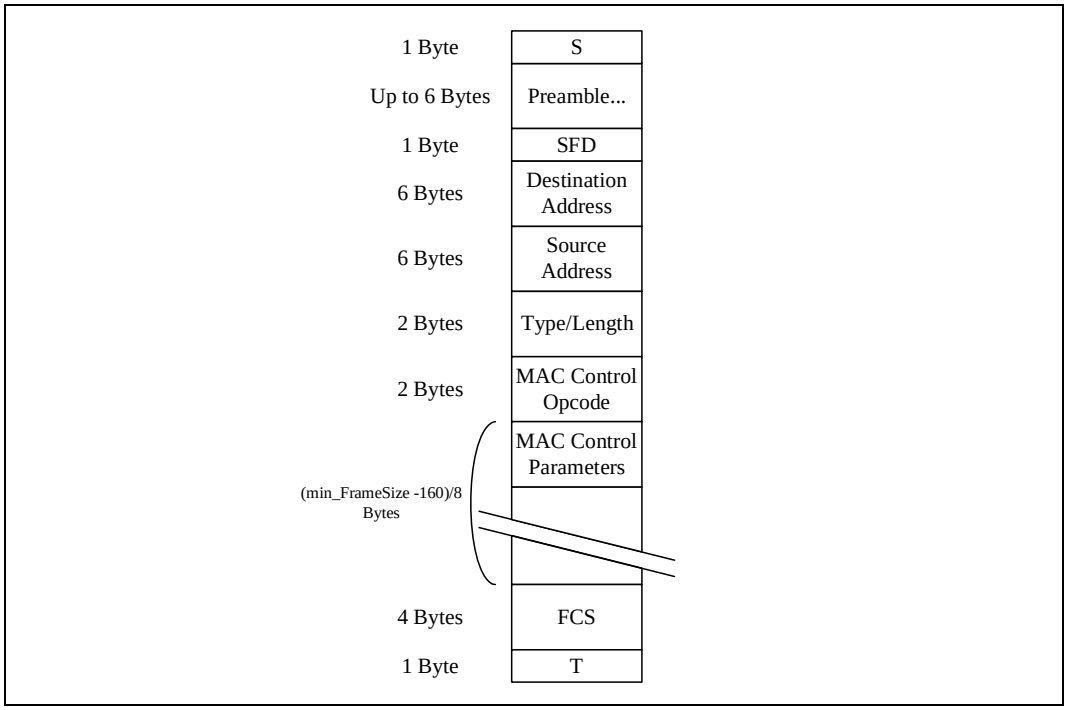
The Flow Control Type Register (FCT) contains a 16 bit field which is compared against the flow control packet's type field to determine if it is a valid flow control packet: XON or XOFF. 802.3x reserves this as 0x8808. This value must be loaded into the FCT.

The final check for a valid PAUSE frame is the MAC Control Opcode. At this time only the PAUSE control frame opcode is defined. It has a value of 0x0001.

Frame based flow control differentiates XOFF from XON based on the value of the PAUSE timer field. Non-zero values constitute XOFF frames while a value of zero constitutes an XON frame. Values in the timer field are in units of slot time. A "slot time" is hard wired to 64 byte times, or 512ns.

Note: An XON frame signals the cancellation of the "pause" initiated by an XOFF frame. "Pause for zero slot times."

Figure 37-47.802.3x MAC Control Frame Format



Where "S" is the Start-of-Packet delimiter and "T" is the first part of the End-of-Packet delimiters for 802.3z encapsulation.

The receiver is enabled to receive flow control frames via the Device Control Register (CTRL.RFCE).

Note: Flow control capability must be negotiated between link partners via the Auto-Negotiation process. The Auto-Negotiation process may modify the value of these bits based on the resolved capability between the local device and the link partner.

Once the receiver has validated the reception of an XOFF, or PAUSE frame, the device will:

- Increment the appropriate statistics register(s)
- Set the STATUS.TXOFF bit
- Initialize the pause timer based on the packet's PAUSE timer field

- Disable packet transmission or schedule the disabling of transmission after the current packet completes.

Resumption of transmission may occur after:

- Expiration of the PAUSE timer
- Reception of on XON frame (a frame with its PAUSE timer set to 0)

Either condition clears the Device Status Register (STATUS.TXOFF) and transmission may resume. Hardware records the number of received XON frames.

37.5.8.5.2 "Discard PAUSE Frames" & "Pass MAC Control Frames"

Two bits in the Receive Control Register are implemented specifically for control over receipt of PAUSE and MAC control frames. These bits are Discard PAUSE Frames (RCTL.DPF) and Pass MAC Control Frames (RCTL.PMCF). See "RCTL – Receive Control Register" on page 1474 for explicit definitions.

RCTL.DPF bit will force the discarding of any valid PAUSE frame addressed to the device's station address. If the packet is a valid PAUSE frame and is addressed to the station address (receive address [0]), the device will not pass the packet to host memory if RCTL.DPF bit is set to logic high. However, if a flow control packet is sent to the station address, and is a valid flow control frame, it will be DMA'd when RCTL.DPF is set to zero. This bit has no affect on PAUSE operation, only the DMA function.

RCTL.PMCF bit allows for the passing of any valid MAC control frames to the system which do not have a valid PAUSE opcode. In other words, the frame must have the correct MAC control frame multicast address (or the MAC station address) as well as the correct type field match with the FCT register, but will not have the defined PAUSE opcode of 0x0001. Frames of this type are DMA'd to host memory when RCTL.PMCF is logic high.

37.5.8.5.3 Transmission of PAUSE Frames

Transmission of PAUSE frames is enabled by software with the Device Control Register (CTRL.TFCE).

Note: Similar to the reception flow control packets mentioned above, XOFF packets may be transmitted only if this configuration has been negotiated between the link partners via the Auto-Negotiation process. In other words, the setting of this bit indicates the desired configuration. The resolution of the Auto-Negotiation process is indicated in "Physical Layer Auto-Negotiation & Link Setup Features" on page 1394.

The content of the Flow Control Receive Threshold High Register determines at what point hardware transmits a PAUSE frame. Hardware monitors the fullness of the receive FIFO and compares it with the contents of FCRTH. When the threshold is reached, hardware sends a PAUSE frame with its pause time field equal to FCTTV register. Once the receive buffer fullness reaches the low water mark, hardware sends an XON message (a PAUSE frame with a timer value of 0). Software enables this capability with the Flow Control Receive Threshold Low Register (FCRTL.XONE).

Hardware will send one more PAUSE frame if it has previously sent one and the FIFO overflows (so the threshold must not be set greater than the FIFO size). This is intended to minimize the amount of packets dropped if the first PAUSE frame does not reach its target. Since the secure receive packets use the same data path, the behavior is identical when secure packets are received.

Note: The transmission of Flow Control frames should only be enabled in full duplex mode per the IEEE 802.3 standard. Software should ensure that the transmission of flow control packets is disabled when the device is operating in half-duplex mode.

37.5.8.5.4 Software Initiated PAUSE Frame Transmission

The GbE has the added capability to transmit an XOFF frame via software. This is accomplished by software using the Transmit Control Register (TCTL.SWXOFF). Once this bit is set, hardware will initiate the transmission of a PAUSE frame in a manner similar to that automatically generated by hardware.

TCTL.SWXOFF is self clearing after the PAUSE frame has been transmitted.

The state of the CTRL.TFCE bit or the negotiated flow control configuration does not affect software generated PAUSE frame transmission.

Note: Software sends an XON frame by programming a zero in the PAUSE timer field of the FCTTV register. XOFF transmission is not supported in 802.3x for half duplex links. Software should not initiate an XOFF or XON transmission if the device is configured for half duplex operation.

37.5.9 802.1q VLAN Support

The GbE provides specific mechanisms to support 802.1q VLANs, namely:

- Optional adding (for transmits) and ping (for receives) of IEEE 802.1q VLAN tags.
- Optional ability to filter packets belonging to certain 802.1q VLANs.

The difference between an untagged 802.3 Ethernet packet and an 802.1q VLAN tagged packet is minimal, as shown in Table 37-4.

Table 37-4. Untagged 802.3 Packet vs 802.1q VLAN tagged Packet

802.3 Packet	Number of Octets
DA	6
SA	6
Type/Length	2
Data	46-1500
CRC	4

802.1q VLAN Packet	Number of Octets
DA	6
SA	6
802.1q Tag	4
Type/Length	2
Data	46-1500
CRC*	4

* The CRC for the 802.1q tagged frame is re-computed, so that it covers the entire tagged frame including the 802.1q tag header. Also, max frame size for an 802.1q VLAN packet is 1522 octets as opposed to 1518 octets for a normal 802.3z Ethernet packet.

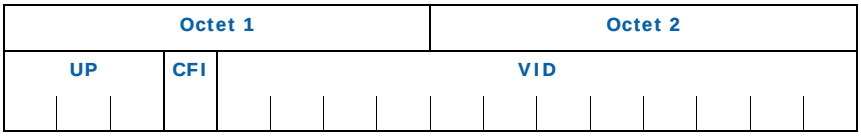
37.5.9.0.1 802.1q Tagged Frames

For 802.1q, the Tag Header field consists of four octets comprised of the Tag Protocol Identifier (TPID) and Tag Control Information (TCI); each taking 2 octets. The first 16 bits of the tag header makes up the TPID. It contains the “protocol type” which identifies the packet as a valid 802.1q tagged packet.

The two octets making up the TCI contain three fields:

- User Priority (UP)
- Canonical Form Indicator (CFI), which should be “0” for transmits. For receives, the device has the capability to filter out packets that have this bit set. Refer to RCTL.CFIEN and RCTL.CFI described in Section 37.6.4.1, “RCTL – Receive Control Register”.
- VLAN Identifier (VID)

Figure 37-48.TCI Bit Ordering



37.5.9.1 Transmitting and Receiving 802.1q Packets

Since the 802.1q tag is only four bytes, adding and stripping of tags could be done completely in software. (i.e., For transmits, software inserts the tag into packet data before it builds the transmit descriptor list, and for receives, software strips the 4 byte tag from the packet data before delivering the packet to upper layer software.)

However, because adding and stripping of tags in software results in more over-head for the host, the GbE has additional capabilities to add and strip tags in hardware.

37.5.9.1.1 Adding 802.1q Tags on Transmits

Software may command the GbE to insert an 802.1q VLAN tag on a per packet basis. If CTRL.VME is set to 1, and the VLE bit in the transmit descriptor is set to 1, then the EP80579's GbE will insert a VLAN tag into the packet that it transmits over the wire. The Tag Protocol Identifier (TPID) field of the 802.1q tag comes from the VET register, and the Tag Control Information (TCI) of the 802.1q tag comes from the special field Figure 37-9, "Special Descriptor Field Layout" on page 1358of the transmit descriptor. Refer to Table 37-2, "VLAN Tag Insertion Decision Table when VLAN Mode Enabled (CTRL.VME=1)" on page 1368, for more information regarding hardware insertion of tags for transmits.

Similarly, software can instruct the GbE to insert an 802.1q VLAN tag for secure packets. Software gives the command "transmit with VLAN" to the security subsystem to instruct hardware that a VLAN tag should be inserted before the packet is sent on the wire. The Tag Protocol Identifier (TPID) field of the 802.1q tag comes from the VET register, and the Tag Control Information (TCI) of the 802.1q tag is specified in the "transmit with VLAN" command itself.

37.5.9.1.2 Stripping 802.1q Tags on Receives

Software may instruct the GbE to strip 802.1q VLAN tags from received packets. If the CTRL.VME bit is set to 1, and the incoming packet is an 802.1q VLAN packet (i.e. it's Ethernet Type field matched the VET register), then the EP80579's GbE strips the 4 byte VLAN tag from the packet, and stores the TCI in the Special field (see Figure 37-9, "Special Descriptor Field Layout" on page 1358) of the receive descriptor.

The EP80579's GbE also sets the VP bit in the receive descriptor to indicate that the packet had a VLAN tag that was stripped. If the CTRL.VME bit is not set, the 802.1Q packets can still be received if they pass the receive filter, but the VLAN tag will not be stripped and the VP bit will not be set. Refer to Table 37-5 on page 1402 for more information regarding receive packet filtering.

37.5.9.2 802.1q VLAN Packet Filtering

VLAN filtering is enabled by setting the RCTL.VFE bit to 1. If enabled, hardware compares the type field of the incoming packet to a 16 bit field in the VLAN EtherType Register (VET). If the VLAN type field in the incoming packet matches the VET register, the packet is then compared against the VLAN Filter Table Array for acceptance.

The Virtual LAN ID field indexes a 4096 bit vector. If the indexed bit in the vector is one; there is a Virtual LAN match. Software may set the entire bit vector to ones if the node does not implement 802.1q filtering. The register description of the VLAN Filter Table Array is described in detail in "VFTA[0-127] – 128 VLAN Filter Table Array Registers" on page 1490.

In summary, the 4096 bit vector is comprised of 128, 32-bit registers. Matching to this bit vector follows the same algorithm as indicated in section 0 for Multicast Address filtering. The VLAN Identifier (VID) field consists of 12 bits. The upper 7 bits of this field are decoded to determine the 32-bit register in the VLAN Filter Table Array to address and the lower 5 bits determine which of the 32 bits in the register to evaluate for matching.

Two other bits RCTL.CFIEN and RCTL.CFI (refer to "RCTL – Receive Control Register" on page 1474), are also used in conjunction with 802.1q VLAN filtering operations. RCTL.CFIEN enables the comparison of the value of the RCTL.CFI bit to the 802.1q packet data as an acceptance criteria for the packet.

Note: The VFE bit does not effect whether the VLAN tag is stripped. It only effects whether the VLAN packet passes the receive filter.

The following table lists reception actions per control bit settings.

Table 37-5. Packet Reception Decision Table

Is packet 802.1q?	CTRL. VME	RCTL. VFE	Action
No	X	X	Normal packet reception
Yes	0	0	Receive a VLAN packet if it passes the standard filters (only). Leave the packet as received in the data buffer. VP bit in receive descriptor is clear.
Yes	0	1	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Leave the packet as received in the data buffer (e.g. the VLAN tag would not be stripped). VP bit in receive descriptor is clear.
Yes	1	0	Receive a VLAN packet if it passes the standard filters (only). Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Set VP bit in receive descriptor.
Yes	1	1	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Set VP bit in receive descriptor.

Note: A packet is defined as a VLAN/802.1q packet if its type field matches the VET.

37.5.10 Wake on LAN

Two types of wakeup mechanisms are supported:

- Advanced Power Management (APM) Wakeup
- ACPI Power Management Wakeup

When so configured, if a wake-up packet is received, the GBE_PME_WAKE signal will be asserted. The GBE_PME_WAKE signal of all three GbE MACs are wired-or together and brought to the external pin GBE_PME_WAKE. The user must externally connect this pin to the PME_N input pin.

37.5.10.1 Advanced Power Management Wakeup

"Advanced Power Management Wakeup", or "APM Wakeup", was previously known as "Wake on LAN". It is an feature that has existed in ethernet NICs for several generations. The basic premise is to receive a broadcast or unicast packet with an

explicit data pattern, and then to assert a signal to wake-up the system. In the earlier generations, this was accomplished by using a special signal that ran across a cable to a defined connector on the motherboard. The NIC would assert the signal for approximately 50ms to signal a wakeup. In more recent implementations, the PCI PME# signal has been used to wake-up the system.

On power-up, the APM Enable bits from the EEPROM Initialization Control Word 2 are read into the APM Enable (APME) bits of the Wakeup Control Register (WCR). These bits control enabling of APM Wakeup.

When APM Wakeup is enabled, the controller checks all incoming packets for “Magic Packets”.

Once a matching magic packet is received, the following occurs:

- If the Assert PME On APM Wakeup (APMPME) bit is set in the Wake Up Control Register (WUCR):
  - — set the PME_Status bit in the Power Management Control / Status Register (PMCSR) and assert GBE_PME_WAKE.
- Set the Magic Packet Received bit in the Wake Up Status Register (WUS).

The controller will assert GBE_PME_WAKE until the driver does one of the following:

- clears the Magic Packet Received AMAG bit in the Wake Up Status Register (WUS)
- clears the Assert PME On APM Wakeup (APMPME) bit in the Wake Up Control Register (WUC)
- disables APM Wakeup.

“APM Wakeup” is supported in all power states and only disabled if a subsequent EEPROM read results in the APM Wake Up bit being cleared or the software explicitly writes a 0 to the APM Wake Up (APM) bit of the WUC register.

37.5.10.2 ACPI Power Management Wakeup

Three sources of ACPI Power Management based Wakeups are supported:

- Reception of a “Magic Packet”.

Reception of a Network Wakeup Packet.

Activating ACPI Power Management Wakeup requires the following steps:

- The driver programs the Wake Up Filter Control Register (WUFC) to indicate the packets it wishes to wake up and supplies the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT) and the Flexible Filter Mask Table (f), Flexible Filter Length Table (FFLT), and the Flexible Filter Value Table (FFVT). The OS writes a 1 to the Pme_En bit of the Power Management Control / Status Register (PMCSR.8).

Normally, after enabling wakeup, the OS will write (11)b to the lower two bits of the PMCSR to put the GbE controller into low-power mode.

Once Wakeup is enabled, the controller monitors incoming packets, first filtering them according to its standard address filtering method, then filtering them with all of the enabled wakeup filters. If a packet passes both the standard address filtering and at least one of the enabled wakeup filters, the controller will:

- Set the PME_Status bit in the Power Management Control / Status Register (PMCSR)
- If the PME_En bit in the Power Management Control / Status Register (PMCSR) is set, assert GBE_PME_WAKE.

- Set one or more of the “Received” bits in the Wake Up Status Register (WUS). (The controller will set more than one bit if a packet matches more than one filter.)

GBE_PME_WAKE will remain asserted until the OS either writes a 1 to the PME_Status bit of the PMCSR register or writes a 0 to the Pme_En bit.

After receiving a wakeup packet, the controller will ignore any subsequent wakeup packets until the driver clears all of the “Received” bits in the Wake Up Status Register (WUS).

37.5.10.3 Wake-up Packets: Pre-defined Filters

Various wakeup packets are supported using two types of filters:

- Pre-defined Filters
- Flexible Filters

Each of these filters will be enabled if the corresponding bit in the Wake Up Filter Control Register (WUFC) is set to 1.

This section describes the functioning of the Pre-defined Filters.

The following packets are supported by the Pre-defined Filters:

- Directed Packet (including exact, multicast indexed, and broadcast)
- Magic Packet
- ARP/IPv4 Request Packet
- Directed IPv4 Packet
- Directed IPv6 Packet

Each of these filters will be enabled if the corresponding bit in the Wakeup Filter Control Register (WUFC) is set to 1.

The explanation of each filter includes a table showing which bytes at which offsets are compared to determine if the packet passes the filter.

VLAN and LLC/SNAP Packets

Various tables may also include a reference to a possible VLAN Tag and LLC/SNAP Header. The controller detects VLAN and LLC/Snap frames by checking the initial size/type field. It first checks for a VLAN header by comparing the size/type field to the value programmed in the VLAN EtherType register. If the field matches then the frame is considered a VLAN frame. It will then check the VLAN ID against the values programmed in the VLAN Filter Table Array. If the ID matches the packet processing continues. If the ID doesn't match, or the CTRL.VME bit is 0, and the VLAN Tag is listed as “Compare” in the table, the packet will not be considered a wakeup packet.

After processing a possible VLAN Tag the controller will check for a LLC/SNAP Header. If the size/type field is less than or equal to 1500 bytes, the controller will check the following 6 bytes for the pattern AA_AA_03_00_00_00. If the pattern matches then the packet processing continues. If the pattern doesn't match, and the LLC/SAP Header is listed as “Compare” or “Check” in the table the packet will not be considered a wakeup packet.

37.5.10.3.1 Directed Exact Packet

The GbE controller will generate a wakeup event upon reception of any packet whose destination address matches one of the 16 valid programmed Receive Addresses if the Directed Exact Wake Up Enable bit is set in the Wake Up Filter Control Register (WUFC.EX).

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address	Programmable	Compare	Match any pre-programmed address

37.5.10.3.2 Directed Multicast Packet

For multicast packets, the upper bits of the incoming packet's destination address index a bit vector in the Multicast Table Array, that indicates whether to accept the packet. If the Directed Multicast Wake Up Enable bit set in the Wake Up Filter Control Register (WUFC.MC) and the indexed bit in the vector is one then the controller will generate a wakeup event. The exact bits used in the comparison are programmed by software in the Multicast Offset field of the Receive Control Register (RCTL.MO).

If the MAC has been configured for promiscuous mode, a multicast wakeup will occur if a broadcast packet is received. This is because a broadcast message is a special type of multicast message. Refer to 802.3, section 3.2.3.1.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	See paragraph

37.5.10.3.3 Broadcast

If the Broadcast Wake Up Enable bit in the Wake Up Filter Control Register (WUFC.BC) is set the controller will generate a wake up event when it receives a broadcast packet.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address	FF * 6	Compare	

37.5.10.3.4 Magic Packet

Magic Packet Technology Details: Once the LAN controller has been put into the Magic Packet mode, it scans all incoming frames addressed to the node for a specific data sequence, which indicates to the controller that this is a Magic Packet frame. A Magic Packet frame must also meet the basic requirements for the LAN technology chosen, such as SOURCE ADDRESS, DESTINATION ADDRESS (which may be the receiving station's IEEE address or a MULTICAST address which includes the BROADCAST address), and CRC. The specific data sequence consists of 16 duplications of the IEEE address of this node, with no breaks or interruptions. This sequence can be located anywhere within the packet, but must be preceded by a synchronization stream. The synchronization stream allows the scanning state machine to be much simpler. The synchronization stream is defined as 6 bytes of FFh. The device will also accept a BROADCAST frame, as long as the 16 duplications of the IEEE address match the address of the machine to be awakened.

The controller will expect the destination address to either:

- Be the broadcast address (FF.FF.FF.FF.FF.FF)
- Match the value in Receive Address Register 0 (RAH0, RAL0). This is initially loaded from the EEPROM but may be changed by the driver.
- Match any other address filtering enabled by the driver.

The controller will search for the contents of Receive Address Register 0 (RAH0, RAL0) as the embedded IEEE address. It will consider any non-FF byte after a series of at least 6 FFs to be the start of the IEEE address for comparison purposes. (I.E. It will catch the case of 7 FFs followed by the IEEE address). As soon as one of the first 96 bytes after a string of FFs doesn't match, it will continue to search for anther set of at least 6 FFs followed by the 16 copies of the IEEE address later in the packet. Note that this definition precludes the first byte of the destination address from being FF.

A Magic Packet's destination address must match the address filtering enabled in the configuration registers with the exception that broadcast packets will be considered to match even if the Broadcast Accept bit of the Receive Control Register (RCTL.BAM) is 0. If APM Wakeup is enabled in the EEPROM, the controller will start up with the Receive Address Register 0 (RAH0, RAL0) loaded from the EEPROM. This is to permit the controller to accept packets with the matching IEEE address before the driver comes up.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header- processed by main address filter
...	...	...		Skip	
Any	6	Synchronizing Stream	FF FF FF FF FF FF +	Compare	
Any + 6	96	16 copies of Node Address	Node Address*16	Compare	Compared to Receive Address Register 0 (RAH0, RAL0)

37.5.10.3.5 ARP / IPv4 Request Packet

The GbE controller will support reception of ARP Request packets for wake up if the ARP bit is set in the Wake Up Filter Control Register (WUFC). Four IPv4 addresses are supported which are programmed in the IPv4 Address Table (IP4AT). A successfully matched packet must contain a broadcast MAC address, a Type of 0x0806, an ARP OPCODE of 0x01, and one of the four programmed IPv4 addresses. The GbE controller also handles ARP Request packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header- processed by main address filter
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag	8100h + check ID	Check	
12+S	D=(0/8)	Possible LLC/ SNAP Header	Type <= 1500 + AAAA03000000h	Check	
12+D+S	2	Type	0806h	Compare	ARP
ARP Header					
14+D+S	2	HW Type	0001h	Compare	

Offset	# of bytes	Field	Value	Action	Comment
16+D+S	2	Protocol Type	0800h	Compare	
18+D+S	1	Hardware Size	06h	Compare	
19+D+S	1	Protocol Address Length	04h	Compare	
20+D+S	2	Operation	0001h	Compare	
22+D+S	6	Sender HW Address	-	Ignore	
28+D+S	4	Sender IP Address	-	Ignore	
32+D+S	6	Target HW Address	-	Ignore	
38+D+S	4	Target IP Address	IP4AT	Compare	May match any of 4 values in IP4AT

37.5.10.3.6 Directed IPv4 Packet

The LAN controller will support reception of Directed IPv4 packets for wake up if the IPV4 bit is set in the Wake Up Filter Control Register (WUFC). Four IPv4 addresses are supported which are programmed in the IPv4 Address Table (IP4AT). A successfully matched packet must contain the station's MAC address, a Type of 0x0800, and one of the four programmed IPv4 addresses. The GbE controller also handles Directed IPv4 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address	-	Compare	MAC Header- processed by main address filter
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag	8100h + check ID	Check	
12 12+S	D=(0/8)	Possible LLC/ SNAP Header	Type <= 1500 + AAAA03000000h	Check	
12 12+D+S	2	Type	0800h	Compare	IP
ARP Header					
14 14+D+S	1	Version(4-bit) / HDR Length(4-bit)	4xh	Compare	Check IPv4 and header length
15 15+D+S	1	Type of Service	-	Ignore	
16 16+D+S	2	Packet Length	-	Ignore	
18 18+D+S	2	Identification	-	Ignore	

Offset	# of bytes	Field	Value	Action	Comment
20 20+D+S	2	Fragment Info	-	Ignore	
22 22+D+S	1	Time to Live	-	Ignore	
23 23+D+S	1	Protocol	-	Ignore	
24 24+D+S	2	Header Checksum	-	Ignore	
26 26+D+S	4	Source IP Address	-	Ignore	
30 30+D+S	4	Destination IP Address	IPv4AT	Compare	May match any of four values in IPv4AT

37.5.10.3.7 Directed IPv6 Packet

The LAN controller will support reception of Directed IPv6 packets for wake up if the IPV6 bit is set in the Wake Up Filter Control Register (WUFC). One IPv6 address is supported and it is programmed in the IPv6 Address Table (IP6AT). A successfully matched packet must contain the station's MAC address, a Type of 0x0800, and the programmed IPv6 address. This MAC also handles Directed IPv6 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header- processed by main address filter
6	6	Source Address		Ignore	
12	S=(0/4)	Possible VLAN Tag	8100h + check ID	Check	
12+S	D=(0/8)	Possible LLC/ SNAP Header	Type <= 1500 + AAAA03000000h	Check	
12+D+S	2	Type	86DDh	Compare	IPv6
ARP Header					
14+D+S	1	Version/ Traffic Class	6Xh	Compare	Check IPv6
16+D+S	3	Traffic Class/Flow Label	-	Ignore	
18+D+S	2	Payload Length	-	Ignore	
20+D+S	1	Next Header	IPv6 next header types	Check	

Offset	# of bytes	Field	Value	Action	Comment
21+D+S	1	Hop Limit	-	Ignore	
22+D+S	16	Source Address	-	Ignore	
38+D+S	16	Destination Address	IP6AT	Compare	Match value in IP6AT

37.5.10.4 Wake-up Packets: Flexible Filters

Various wakeup packets are supported using two types of filters:

- Pre-defined Filters
- Flexible Filters

Each of these filters will be enabled if the corresponding bit in the Wake Up Filter Control Register (WUFC) is set to 1.

This section describes the functioning of Flexible Filters.

A total of four flexible filters are supported. Each filter is can be configured to recognize any arbitrary pattern within the first 128 byte of the packet To configure the flexible filter, the software programs the mask values into the Flexible Filter Mask Table (FFMT) and the required values into the Flexible Filter Value Table (FFVT), and the minimum packet length into the Flexible Filter Length Table (FFLT). These contain separate values for each filter. The software must also enable the filter in the Wake Up Filter Control Register (WUFC), and enable the overall wake up functionality must be enabled by setting PME_En in the Power Management Control Status Register or the Wake Up Control Register.

Once enabled, the flexible filters will scan incoming packets for a match. If the filter encounters any byte in the packet where the mask bit is one and the byte doesn't match the byte programmed in the Flexible Filter Value Table (FFVT) then the filter will fail that packet. If the filter reaches the required length without failing the packet, it passes the packet and generates a wake up event. It will ignore any mask bits set to one beyond the required length.

The flexible filter does not have any way to automatically skip VLAN or LLC/SNAP headers. If such headers are included the offsets of the subsequent fields must be adjusted accordingly.

The following packets are listed for reference purposes only. The flexible filter could be used to filter these packets.

37.5.10.4.1 IPX Diagnostic Responder Request Packet

An IPX Diagnostic Responder Request Packet must contain a valid MAC address, a Protocol Type of 0x8137, and an IPX Diagnostic Socket of 0x0456. It may include LLC/SNAP Headers and VLAN Tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP Headers and VLAN tags.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Compare or Skip	
12+S	D=(0/8)	Possible LLC/ SNAP Header		Compare or Skip	
12+D+S	2	Type	8137h	Compare	IPX
IPX Header/Info					
14+D+S	16	Some IPX Stuff	-	Ignore	
30+D+S	2	IPX Diagnostic Socket	0x0456	Compare	

37.5.10.4.2 Directed IPX Packet

A valid Directed IPX Packet contain the station's MAC address, a Protocol Type of 0x8137, and an IPX Node Address that equals to the station's MAC address. It may include LLC/SNAP Headers and VLAN Tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP Headers and VLAN tags.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Compare or Skip	
12+S	D=(0/8)	Possible LLC/ SNAP Header		Compare or Skip	
12+D+S	2	Type	8137h	Compare	IPX
IPX Header/Info					
14+D+S	16	Some IPX Stuff	-	Ignore	
24+D+S	2	IPX Node Address	Receive Address 0	Compare	

37.5.10.4.3 IPv6 Neighbor Discovery Filter

In IPv6, a Neighbor Discovery packet is used for address resolution. A flexible filter can be used to check for a “Neighborhood Discovery Packet”.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header - processed by main address filter, or broadcast
6	6	Source Address		Skip	
12	4	Possible VLAN Tag		See text	
12+S	8	Possible LLC/ SNAP Header		See text	
12+D+S	2	Type	86DDh	Compare	IP
IPv6 Header					
14+D+S	1	Version/Traffic Class	6X	Compare	Check IPv6
15+D+S	3	Traffic Class/Flow Label	-	Ignore	
18+D+S	2	Payload Length	-	Ignore	
20+D+S	1	Next Header	3Ah, 00h, 2Bh, or 3Ch	Check	ICMP, or IPv6 next headers: + routing (2Bh) + dest options (3Ch) + hop-by-hop (00h)
21+D+S	1	Hop Limit	FFh	Check	
22+D+S	16	Source IP Address	-	Ignore	
38+D+S	16	Destination IP Address	-	Ignore	
54+D+S	N	Possible IPv6 Next Headers	-	Check	Process headers to get next header. Header type must be routing, destination options, or hop-by-hop.
ICMP Header					
54+D+S+N	1	Type	87h	Check	Neighbor solicitation
55+D+S	1	Code	00h	Check	
56+D+S	2	ICMP Header Checksum		Ignore	
Neighbor Discover Info					
58+D+S+N	4	Reserved	-	Ignore	
62+D+S+N	16	Target Address	-	Check	Match IPV6AT[0]
78+D+S+N	N	Possible source link-layer address	-	Ignore	
...	any	-	-	Ignore	Packet data
58+D+S+N	4	CRC	-	Compare	Validate correct

37.5.11 Serial EEPROM

The GbE controller uses an EEPROM device for storing product configuration information. The EEPROM is divided into two general regions:

- Hardware accessed - loaded by GbE controller after power-up, Unit Reset deassertion, D3>D0 transition, or software commanded EEPROM read (CTRL_EXT.EE_RST).
- Software accessed - used by software only. The meaning of these registers as listed here is a convention for the software only and is ignored by the controller.

37.5.11.1 EEPROM Device

The EEPROM interface supports a MicroWire* interface. It expects the EEPROM to be capable of 1MHz operation.

The EP80579 is compatible with a 4096 bit 4-wire serial EEPROM such as is compatible with a NM93C66 EEPROM. This EEPROM is accessed in 16-bit words, containing 256 words.

The EP80579 will automatically determine whether an EEPROM is connected.

Note: The EP80579 will ONLY detect presence of the EEPROM during power-up.

37.5.11.2 Software Accesses

The MAC provides two different methods for software access to the EEPROM. It can either use the built-in controller to read the EEPROM, or access the EEPROM directly using the EEPROM's 4-wire interface.

Software can use the EEPROM Read register (EERD) to cause the GbE controller to read a word from the EEPROM that the software can then use. To do this, software writes the address to read to the Read Address (EERD.ADDR) field and simultaneously writes a 1 to the Start Read bit (EERD.START). The controller will read the word from the EEPROM, set the Read Done bit (EERD.DONE), and put the data in the Read Data field (EERD.DATA). Software can poll the EEPROM Read register until it sees the Read Done bit set, then use the data from the Read Data field. Any words read this way are not written to internal registers.

Software can also directly access the EEPROM's 4-wire interface through the EEPROM Control Register (EEC). It can use this for reads, writes, or other EEPROM operations.

To directly access the EEPROM, software should follow these steps:

- Write a 1 to the EEPROM Request bit (EEC.EE_REQ)
- Read the EEPROM Grant bit (EEC.EE_GNT) until it becomes 1. It will remain 0 as long as the hardware is accessing the EEPROM.
- Write or read the EEPROM using the direct access to the 4-wire interface as defined in the EEPROM Control & Data Register (EEC). The exact protocol used depends on the EEPROM placed on the board and can be found in the appropriate datasheet.
- Write a 0 to the EEPROM Request bit (EEC.EE_REQ).

Finally, software can cause the controller to re-read the hardware accessed fields of the EEPROM (setting the MAC's internal registers appropriately) by writing a 1 to the EEPROM Reset bit of the Extended Device Control Register (CTRL_EXT.EE_RST).

Note: This will only cause the EEPROM to be re-read. The Gbe will NOT attempt to re-detect the presence of an EEPROM.

37.5.11.3 Signature Field

The only way the MAC has to tell if an EEPROM is present is by trying to read the EEPROM. The MAC will first read the Initialization Control Word 1. It will check the received value for bits 15 and 14. If bit 15 is 0 and bit 14 is 1, it considers the EEPROM to be present and valid and will read additional EEPROM words and program its internal registers based on the values read. Otherwise, it will ignore the values it read from the Initialization Control Word 1 and not read any other words.

37.5.11.4 EEPROM Map

Note: If any one of the GbE interfaces is disabled then the corresponding memory locations in the EEPROM for that device must be initialized to all '0's.

The following table indicates the EEPROM map to be used.

Table 37-6. EEPROM Address Map

Word Offset	Used By MAC	Upper Byte - Bits 15:8	Lower Byte - Bits 7:0
00h	0,1,2	Init Control Word 1	
01h			
02h			
03h	S/W use only	Compatibility High	Compatibility Low
04h	S/W use only	Compatibility High	Compatibility Low
05h	S/W use only	Compatibility High	Compatibility Low
06h	S/W use only	Compatibility High	Compatibility Low
07h	S/W use only	Compatibility High	Compatibility Low
08h	S/W use only	PBA, byte 1	PBA, byte 2
09h	S/W use only	PBA, byte 3	PBA, byte 4
0Ah			
0Bh			
0Ch			
0Dh			
0Eh			
0Fh			
10h	0	Management Control	
11h	0	Init Control Word 2	Init Control Word 3
12h	0	IA Byte 2	IA Byte 1
13h	0	IA Byte 4	IA Byte 3
14h	0	IA Byte 6	IA Byte 5
15h	0	IPv4 Byte 2	IPv4 Byte 1
16h	0	IPv4 Byte 4	IPv4 Byte 3
17h	0	IPv6 Byte 2	IPv6 Byte 1
18h	0	IPv6 Byte 4	IPv6 Byte 3
19h	0	IPv6 Byte 6	IPv6 Byte 5
1Ah	0	IPv6 Byte 8	IPv6 Byte 7
1Bh	0	IPv6 Byte 10	IPv6 Byte 9
1Ch	0	IPv6 Byte 12	IPv6 Byte 11

Table 37-6. EEPROM Address Map

Word Offset	Used By MAC	Upper Byte - Bits 15:8	Lower Byte - Bits 7:0
1Dh	0	IPv6 Byte 14	IPv6 Byte 13
1Eh	0	IPv6 Byte 16	IPv6 Byte 15
1Fh			
20h	1	Management Control	
21h	1	Init Control Word 2	Init Control Word 3
22h	1	IA Byte 2	IA Byte 1
23h	1	IA Byte 4	IA Byte 3
24h	1	IA Byte 6	IA Byte 5
25h	1	IPv4 Byte 2	IPv4 Byte 1
26h	1	IPv4 Byte 4	IPv4 Byte 3
27h	1	IPv6 Byte 2	IPv6 Byte 1
28h	1	IPv6 Byte 4	IPv6 Byte 3
29h	1	IPv6 Byte 6	IPv6 Byte 5
2Ah	1	IPv6 Byte 8	IPv6 Byte 7
2Bh	1	IPv6 Byte 10	IPv6 Byte 9
2Ch	1	IPv6 Byte 12	IPv6 Byte 11
2Dh	1	IPv6 Byte 14	IPv6 Byte 13
2Eh	1	IPv6 Byte 16	IPv6 Byte 15
2Fh			
30h	2	Management Control	
31h	2	Init Control Word 2	Init Control Word3
32h	2	IA Byte 2	IA Byte 1
33h	2	IA Byte 4	IA Byte 3
34h	2	IA Byte 6	IA Byte 5
35h	2	IPv4 Byte 2	IPv4 Byte 1
36h	2	IPv4 Byte 4	IPv4 Byte 3
37h	2	IPv6 Byte 2	IPv6 Byte 1
38h	2	IPv6 Byte 4	IPv6 Byte 3
39h	2	IPv6 Byte 6	IPv6 Byte 5
3Ah	2	IPv6 Byte 8	IPv6 Byte 7
3Bh	2	IPv6 Byte 10	IPv6 Byte 9
3Ch	2	IPv6 Byte 12	IPv6 Byte 11
3Dh	2	IPv6 Byte 14	IPv6 Byte 13
3Eh	2	IPv6 Byte 16	IPv6 Byte 15
3Fh	S/W use only	Software Checksum	
40h	S/W use only	PXE Word 0 (Software Use) Config	
41h	S/W use only	PXE Word 1 (Software Use) Config	
42h	S/W use only	PXE Word (Software Use) PXE Version	
43h	S/W use only	PXE Word (Software Use) EFI Version	
44h	S/W use only	PXE Word	

Table 37-6. EEPROM Address Map

Word Offset	Used By MAC	Upper Byte - Bits 15:8	Lower Byte - Bits 7:0
45h	S/W use only	PXE Word	
46h	S/W use only	PXE Word	
47h	S/W use only	PXE Word	
48h	S/W use only	PXE Word	
49h	S/W use only	PXE Word	
4Ah	S/W use only	PXE Word	
4Bh	S/W use only	PXE Word	
4Ch	S/W use only	PXE Word	
4Eh	S/W use only	PXE Word	
4Fh		Reserved	
50h . . FFh		Available for Software	

37.5.11.5 Hardware Accessed Words

This section describes the EEPROM words that are loaded and used by the MAC. Most of these bits are located into a configuration registers. The words will only be read and used if the signature field in the Initialization Control Word 1 field correct.

37.5.11.5.1 Initialization Control Word 1

The first word read by the MAC and common for all MACs.

Table 37-7. Initialization Control Word 1

Bit	Bit Acronym	Bit Description
15:14	Signature	The Signature field is a signature of (01)b, indicating to the device that there is a valid EEPROM present. If the Signature field is not (01)b, the other bits in this word are ignored, no further EEPROM read is performed, and default values are used for the configuration space IDs.
13:12	Reserved	Reserved
11	Reserved	Reserved. Must be set to '1'
10	FD	**Full Duplex.** Controls the MAC duplex setting. 0 = Half Duplex 1 = Full Duplex In half-duplex mode, the EP80579's GbE transmits carrier extended packets and can receive both carrier extended packets, and packets transmitted with bursting. This bit overrides the in register CTRL.FD
9	Reserved	Reserved. Must be set to '1'
8:4	Reserved	Reserved
3	Power Management	0 - The Power Management Registers set is read only, and the MAC will not execute a hardware transition to D3. 1 - Full support for power management
2:0	Reserved	Reserved

37.5.11.5.2 Initialization Control Word 2

Each MAC has a unique Initialization Control Word 2.

Table 37-8. Initialization Control Word 2

Bit Range	Bit Acronym	Bit Description
7	APM PME# Enable	Initial value of the *Assert PME On APM Wakeup* bit in the *Wake Up Control Register* (WUC.APMPME).
6:0	Reserved	These bits must be set to '0'

37.5.11.5.3 Initialization Control Word 3

Each MAC has a unique Initialization Control Word 3.

Table 37-9. Initialization Control Word 3

Bit Range	Bit Acronym	Bit Description
7	D3COLD_WAKEUP_ _ADV_EN	Configures the initial HW default value of the ADVD3WUC bit in the Device Control Register (CTRL) following powerup
6:5	Reserved	This bit must be set to '0'
4	RGMII/RMII	RGMII/RMII Translation Gasket Select • '0' - RGMII • '1' - RMII • Initializes CTRL_AUX.RGMII/RMII bit
3	Reserved	This bit must be set to '0'
2	APM Enable	Initial value of *Advanced Power Management Wake Up Enable* in the *Wake Up Control Register* (WUC.APME). This bit is also used to initialize the SLU bit in the CTRL register.
1:0	Link Mode	Initial value of Link Mode bits of the Extended Device Control Register (CTRL_EXT.LINK_MODE), specifying which link interface and protocol is used by the MAC. 00 = MAC operates in GMII/MII mode 01 = Reserved 10 = Reserved 11 = Reserved

37.5.11.5.4 Management Control Word

This register is unique for each MAC.

Table 37-10. Management Control Word

Bit	Bit Acronym	Bit Description
15:13	Reserved	Reserved. Set to 0.
12:8	Reserved	Reserved. Set to 0.
7	IPv6 Address Valid	IPv6 Address in the *IP Address* EEPROM register is valid. This is written to bit 16 of the *IP Address Valid* (IPAV[16]) register.
6	IPv4 Address Valid	IPv4 Address in the *IP Address* EEPROM register is valid. This is written to bit 0 of the *IP Address Valid* (IPAV[0]) register.
5:0	Reserved	Reserved. Set to 0.

37.5.11.5.5 Ethernet Address

The Ethernet Individual Address (IA) is a 6 byte field that must be unique for each adapter card, and thus unique for each MAC. The first three bytes define a group of Individual Addresses which are allocated to a specific vendor - for example, [00 AA 00] or [E9 07 00] was allocated for certain Intel products. The remaining three-bytes are allocated in manufacturing to form a unique value for each EEPROM image. These values change regularly as new products are introduced and/or allocated during in production. The value from this EEPROM field is loaded into the Receive Address Register 0 (RAL0/RAH0).

For the purpose of this specification, the IA byte numbering convention is indicated below:

	IA Byte / Value					
	1	2	3	4	5	6
Field	vendor-specific	vendor-specific	vendor-specific	unique	unique	unique
Example	E9	07	00	variable	variable	variable

37.5.11.5.6 IPv4 Address

Table 37-11. IPv4 Address

Bit	Name	Bit Description
31:0	IPv4 Address	The initial value of IP4AT Address Table entry 0. (IP4AT[0]) See the EEPROM map for an indication of how the bytes are stored.

37.5.11.5.7 IPv6 Address

Table 37-12. IPv6 Address

Bit	Name	Bit Description
127:0	IPv6 Address	The initial value of IPv6 Address Table entry 0. (IP6AT[0]) See the EEPROM map for an indication of how the bytes are stored.

37.5.11.6 Software Accessed Words

37.5.11.6.1 Compatibility Fields

Five words in the EEPROM image are reserved for compatibility information. Hardware does not use these fields or impose any restrictions on their use.

37.5.11.6.2 PBA Number

Two words in the EEPROM image are reserved for PBA information. Hardware does not use these fields or impose any restrictions on their use.

37.5.11.6.3 Checksum Word Calculation

The Checksum word (3Fh) is used to ensure that the base EEPROM image is a valid image. The value of this word should be calculated by adding all the words (00h-3Fh), including the Checksum word itself.

Note: Hardware does not calculate the word 3Fh checksum during EEPROM write; it must be calculated by software independently and included in the EEPROM write data. Hardware does not compute a checksum over words 00h-3Fh during EEPROM reads in order to determine validity of the EEPROM image; this field is provided strictly for SW verification of EEPROM validity. All hardware configuration based on word 00h-3Fh content is based on the validity of the Signature field of EEPROM Initialization Control Word 1 (Signature must be 01b).

37.5.12 Error Handling

The overall principles and goals of error handling for the EP80579 are covered in Chapter 5.0, "Error Handling". This section covers the specifics of error handling including bus errors and soft errors for this unit.

37.5.12.1 CSR (Target) Accesses

The following conditions must be met for a CSR access to be considered valid:

- Command must be CSR read or CSR write
- Length must be 0 (e.g.-4 bytes)
- Byte mask must be 0x0F, 0xF0, or 0x00

If the command violates any of the above conditions then the ICR.ERR_INTBUS condition is asserted:

- All unsupported command types (i.e. those other than CSR read or CSR write) are undefined operations and will result in indeterminate behavior by the GbE target.
- All single CSR writes are assumed to be 32-bit and will complete normally regardless of the byte mask value. Writes using an unsupported byte mask will still be flagged as an error as currently documented but will nonetheless be processed as a 32-bit write
- Read and Write burst operations (length > 0) will be completed on the push/pull buses as single operations. Reads will return 0xFFFF_FFFF for data and will assert the push_data_err signal to the requestor. Write operations will discard the pull data and the requested CSR or memory write will not be performed.
- If the data source of a CSR write operation asserts the internal_bus_data_error signal, then the GbE target discards the data and the write operation is not performed.

The generation of the ICR.ERR_INTBUS condition causes the INTBUS_ERR_STAT register to be updated.

37.5.12.2 DMA Host (Master) Accesses

If the GbE is a master and receives a Push Data Error during a read transaction, then an ICR.ERR_INTBUS condition is generated and the INTBUS_ERR_STAT register is updated. All transactions are terminated.

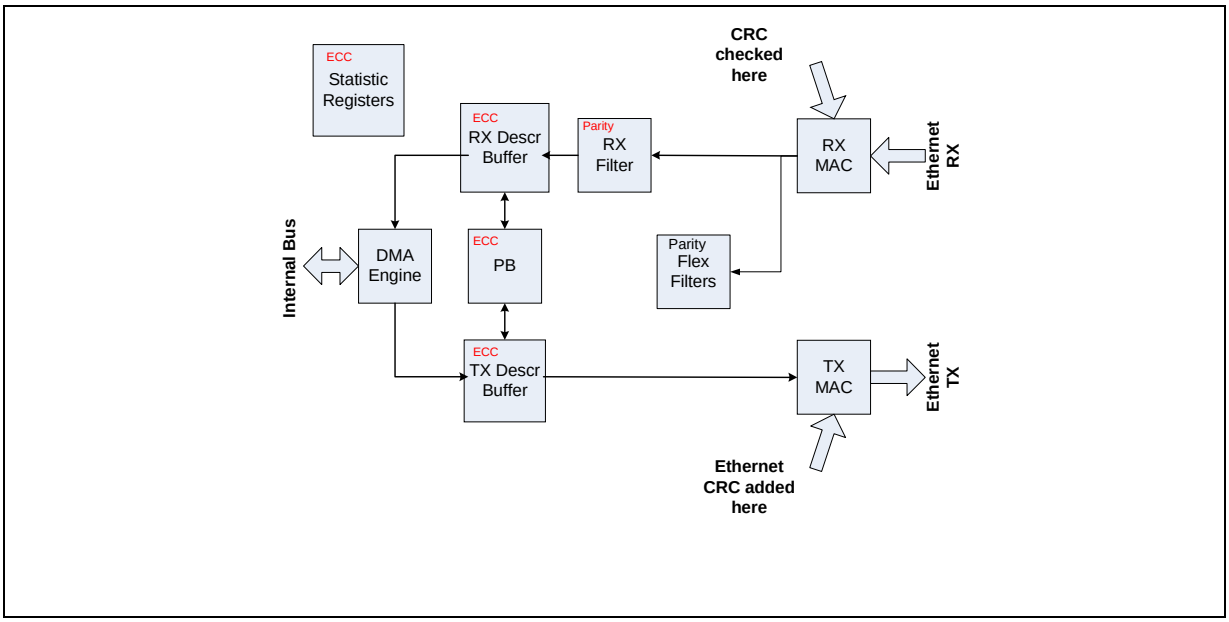
This condition is considered fatal and further Host transaction requests from the GbE unit are inhibited, until a soft reset is issued.

37.5.12.3 Internal Memories

The internal memories are protected via various means to reduce the SER associated with this unit. All memories must be scrubbed to ensure proper operation.

The GbE ECC/parity protected memories indicated in Figure 37-49 may be tested with the Memory Error Test Register (MET). The register will force an error to be written into the memory area selected with the MET.SELECT field. The MET.MASK field will be XOR'd with the ECC or parity bits to insert a parity error into the selected memory, which will cause an error when the memory is read at a later time. Errors will be reported in the appropriate bits of the three ICR registers. Functional or Error interrupt(s) may be generated if enabled.

Figure 37-49. Memory Protection in the GbE



The specifics of the memory protection for each of the memories is shown in Table 37-13, "Memory protection" on page 1419.

Table 37-13. Memory protection

Memory	Size	Protection Type	Protection Bits	Description
Statistic Registers	64 x 32	ECC	8	8-bit ECC code is computed on 32-bit data padded with 0s out to 64 bits for a total of 8 ECC bits (64x40)
RX Filters	128 x 32	Parity	4	parity computed on each byte for a total of 4 parity bits (128x36)
RX Flex Filters	128 x 36	Parity	4	parity computed on each of 4 of the Filter Value and on 4 bits of the Filter Mask for a total of 5 parity bits (128x41)

Table 37-13. Memory protection

Memory	Size	Protection Type	Protection Bits	Description
DMA Tx Descriptor Buffer	64 x 128	ECC	16	8-bit ECC code is computed on each of the upper and lower 64-bits of the 128-bit data for a total of 16 ECC bits (64x144)
DMA Rx Descriptor Buffer	64 x 128	ECC	16	8-bit ECC code is computed on each of the upper and lower 64-bits of the 128-bit data for a total of 16 ECC bits (64x144)
DMA Packet Buffer	4K x 128	ECC	16	8-bit ECC code is computed on each of the upper and lower 64-bits of the 128-bit data for a total of 16 ECC bits (4kx144)

Statistic Registers: In the event of a multi-bit ECC error, an interrupt event will be generated in ICRx.ERR_STAT and the GbE unit will stop all Host Transmit and Receive access to memory, as well as inhibit GbE transmit data. The GbE unit will still remain accessible through the GbE Target interface. It is a software task to detect the event, issue the soft reset to clear the ICRx.ERR_STAT bit and decide what to do with the erroneous data from the statistics register.

RX Filters: In the event of an error from either the multicast filter memory or the special packets filter memory the packet will be rejected, appropriate Statistic Registers will be updated, and an ICRx.ERR_MCFSPF event will be generated. Further GbE DMA Read/Write transactions will not be inhibited, nor will transmit traffic be inhibited.

RX Flex Filters: In the event of a parity error from either any of the flex filter memories the appropriate Statistic Registers will be updated, and an ICRx.ERR_MCFSPF event will be generated. Additionally, if Wake On LAN filtering is enabled, a PME_WAKE event will be generated.

DMA TX Descriptor Buffer: In the event of an ECC error, a ICRx.ERR_TXDS event will be generated and the GbE unit will stop all Host Transmit and Receive access to memory, as well as inhibit GbE transmit data. The GbE unit will still remain accessible through the GbE Target interface. The error indicates the descriptor memory has become untrustworthy and Host accesses are stopped to avoid corruption of other interfaces. It is a software task to detect the event and issue the required soft reset for recovery.

DMA RX Descriptor Buffer: In the event of an ECC error, a ICRx.ERR_RXDS event will be generated and the GbE unit will stop all Host Transmit and Receive access to memory, as well as inhibit GbE transmit data. The GbE unit will still remain accessible through the GbE Target interface. The error indicates the descriptor memory has become untrustworthy and Host accesses are stopped to avoid corruption of other interfaces. It is a software task to detect the event and issue the required soft reset for recovery.

DMA Packet Buffer during Transmit or Receive: In the event of an ECC error, a ICRx.ERR_PB event will be generated and the GbE unit will stop all Host Transmit and Receive access to memory, as well as inhibit GbE transmit data. The GbE unit will still remain accessible through the GbE Target interface. The error indicates the packet buffer memory has become untrustworthy and Host accesses are stopped to avoid corruption of other interfaces. It is a software task to detect the event and issue the required soft reset for recovery.

37.5.13 Reset Operation

The GbE implements multiple hardware and software-initiated reset mechanisms which should be understood and distinguished:

- **SYS_PWR_OK Reset** - The **SYS_PWR_OK** pin acts as a master reset of the entire chip. It is level sensitive, and while it is 0 will hold all HW registers (state and configuration) in reset. **SYS_PWR_OK** is interpreted to be an indication that device power supplies are all stable and the reference clock input is stable.
- **Unit reset**/RESET_N - This pin indicates the reset state of the internal bus. When asserted, the EP80579 will force all GbE output signals to an inactive state. Upon deassertion, the GbE will generate an internal hardware reset, clearing all states, and resetting all configuration registers to HW defaults as specified in "Registers Overview" on page 1425.
- **D3->D0 transition** - This is also known as ACPI Reset. The GbE generates an internal hardware reset on the transition from D3 power state to D0, which clears all and resets nearly all configuration to HW defaults as specified in this document.
- **Soft Reset** - Software can reset the GbE by writing the "CTRL – Device Control Register" on page 1438 (CTRL.RST). A soft reset clears all states and resets nearly all device control registers to HW defaults as specified in "Registers Overview" on page 1425. NOTE: Software must wait a minimum of 5 ms after initiating a soft-reset before accessing any CSR.
- **EEPROM Reset** - Writing a 1 to the EEPROM Reset bit of the Extended Device Control Register (CTRL_EXT.EE_RST) will cause the Gbe to re-read the EEPROM, setting the appropriate bits in the MAC registers whose HW default values are derived from the EEPROM.

The various resets affect the following registers and logic:

Table 37-14. GbE Reset Effects

		Reset Operation						
		PWR_Good Reset	Unit reset	D3-> D0	Soft (CTRL.RST)	Link (LOS deassert)	EEPROM	Notes
Effects to:	Enable / Disable LAN	X	X					
	Tri-State output pins	X	X					
	Data Path	X	X	X	X			
	Configuration Registers	X	X	X	X	X		3
	Read EEPROM	X	X	X	X		4	
	Wake Up Context	X	1	1	1			5
	Wake Up Control Register	X						
	Wake Up Status Registers	X						

1. If AUX_POWER=0 the Wakeup Context is reset
2. N/A
3. The Configuration Registers include:
   - General Registers
   - Interrupt Registers
   - Receive Registers

— Transmit Registers

— Statistics Registers

— Diagnostic Registers

Of these registers, RAH/RAL, MTA[127:0], VFTA[127:0], TDBAH/TDBAL, and RDBAH/RDBAL registers have no default value. If the functions associated with the registers are enabled they must be programmed by software. Once programmed, their value is preserved through all resets as long as power is applied to the bE.

4. See explanation of “EEPROM Resets” above
5. The Wake Up Context is defined in the PCI Bus Power Management Interface Specifications. If includes:

   — PME_En bit of the Power Management Control/Status Register (PMCSR)

   — PME_Status bit of the Power Management Control/Status Register (PMCSR)

   — The shadow copies of these bits in the Wakeup Control Register are treated identically.

Note: In situations where the device is reset using CTRL.RST, the TX data lines will be forced to all zeros. This will cause a substantial number of symbol errors to be detected by the link partner.

37.5.13.1 Soft Reset

This section describes software considerations when using the Soft Reset.

A Soft Reset operation is invoked by software via the Device Control Register (CTRL.RST). The internal hardware reset is delayed until the internal bus is observed to be idle, in order to ensure that no hardware bus protocols are violated. For the next 5 msec, the GbE component is undergoing a low-level hardware reset and re-read of EEPROM in order to re-establish HW defaults. During this time, the GbE will not respond to accesses to the device for a duration of approximately 5 msec.

Note: Software MUST NOT access the GbE device for a minimum of 5 msec after writing the soft-reset bit. Failure to do so may result in a system hang condition. An explicit software wait of 5 milliseconds is necessary to ensure that the device has completed its reset and will respond again.

CTRL.RST self clears upon completion of the reset operation. While the above required wait interval is necessary to ensure that the reset operation is complete, a read of the Device Control Register may be done after the wait interval to confirm the completion of the reset.

37.5.13.2 MAC Disable

This feature allows the MAC and the I/O pins to be put into a very low-power mode. The intended usage of this feature is when the GbE will not be used in the system (no PHY connected).

When the GbE is disabled, all internal clocks are disabled and the GbE is held in reset. The device does not respond to internal bus transactions. Effectively, the unit becomes invisible to the system.

37.5.14 Endianness

Bytes for a receive packet arrive in the order shown from left to right: DA0 DA1 DA2 DA3 DA4 DA5 SA0 SA1 and so on. If the data were to be written to memory in the order of arrival with the first byte written to the lowest address and each subsequent

byte written at the next higher address then the data would appear in memory as shown in Table 37-15, “Long Word Little Endian, Byte Little Endian Ordering” on page 1423.

Table 37-15. Long Word Little Endian, Byte Little Endian Ordering

Byte Address	Bit 63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0	SA1	SA0	DA5	DA4	DA3	DA2	DA1	DA0
8	TOS	45	00	08	SA5	SA4	SA3	SA2
16	06	TTL	Frag LSB	F/F MSB	ID LSB	ID MSB	Len LSB	Len MSB
24	IPDA1	IPDA0	IPSA3	IPSA2	IPSA1	IPSA0	HCS LSB	HCS MSB
32	PB5	PB4	PB3	PB2	PB1	PB0	IPDA3	IPDA2

To allow flexibility in software models, two bits are provided to allow the endianness swizzling described below to be applied to either the descriptor data or the packet data or both or neither. These bits are also located in The CTRL_AUX – Auxiliary Device Control/Status Register. The modes corresponding to these two bits is shown below:

Table 37-16. Endianness Control for Gigabit Ethernet MACs

CSR value	Meaning
00	LW Little-Endian, Byte Big-Endian (Table 37-17)
01	LW Little-Endian, Byte Little-Endian (Table 37-18)
10	LW Big-Endian, Byte Big-Endian (Table 37-19)
11	LW Big-Endian, Byte Little-Endian (Table 37-20)

Table 37-17. Endianness Mode 0: Long Word Little Endian, Byte Big Endian

Byte Address	Bit 63:56	Bit 55:48	Bit 47:40	Bit 39:32	Bit 31:24	Bit 23:16	Bit 15:8	Bit 7:0
0	DA4	DA5	SA0	SA1	DA0	DA1	DA2	DA3
8	08	00	45	TOS	SA2	SA3	SA4	SA5
16	F/F MSB	Frag LSB	TTL	06	Len MSB	Len LSB	ID MSB	ID LSB
24	IPSA2	IPSA3	IPDA0	IPDA1	HCS MSB	HCS LSB	IPSA0	IPSA1
32	PB2	PB3	PB4	PB5	IPDA2	IPDA3	PB0	PB1

Table 37-18. Endianness Mode 1: Long Word Little Endian, Byte Little Endian (Default)

Byte Address	Bit 63:56	Bit 55:48	Bit 47:40	Bit 39:32	Bit 31:24	Bit 23:16	Bit 15:8	Bit 7:0
0	SA1	SA0	DA5	DA4	DA3	DA2	DA1	DA0
8	TOS	45	00	08	SA5	SA4	SA3	SA2
16	06	TTL	Frag LSB	F/F MSB	ID LSB	ID MSB	Len LSB	Len MSB
24	IPDA1	IPDA0	IPSA3	IPSA2	IPSA1	IPSA0	HCS LSB	HCS MSB
32	PB5	PB4	PB3	PB2	PB1	PB0	IPDA3	IPDA2

Table 37-19. Endianness Mode 2: Long Word Big Endian, Byte Big Endian

Byte Address	Bit 63:56	Bit 55:48	Bit 47:40	Bit 39:32	Bit 31:24	Bit 23:16	Bit 15:8	Bit 7:0
0	DA0	DA1	DA2	DA3	DA4	DA5	SA0	SA1
8	SA2	SA3	SA4	SA5	08	00	45	TOS
16	Len MSB	Len LSB	ID MSB	ID LSB	F/F MSB	Frag LSB	TTL	06
24	HCS MSB	HCS LSB	IPSA0	IPSA1	IPSA2	IPSA3	IPDA0	IPDA1
32	IPDA2	IPDA3	PB0	PB1	PB2	PB3	PB4	PB5

Table 37-20. Endianness Mode 3: Long Word Big Endian, Byte Little Endian

Byte Address	Bit 63:56	Bit 55:48	Bit 47:40	Bit 39:32	Bit 31:24	Bit 23:16	Bit 15:8	Bit 7:0
0	DA3	DA2	DA1	DA0	SA1	SA0	DA5	DA4
8	SA5	SA4	SA3	SA2	TOS	45	00	08
16	ID LSB	ID MSB	Len LSB	Len MSB	06	TTL	Frag LSB	F/F MSB
24	IPSA1	IPSA0	HCS LSB	HCS MSB	IPDA1	IPDA0	IPSA3	IPSA2
32	PB1	PB0	IPDA3	IPDA2	PB5	PB4	PB3	PB2

37.6 GbE Controller Register Summary

This section details the programmer-visible state inside the GbE controller. In some cases, it describes hardware structures invisible to software in order to clarify a concept.

The GbE address space is directly memory-mapped to 128 Kbyte of internal registers and memories. This register and memory space is divided into the following categories:

- General
- Receive
- Transmit
- Statistics
- Diagnostics (including packet buffer memory access)
- The external PHY registers are accessed through the MDIO interface. (For more information on MDIO, see “Global Configuration Unit”.)

37.6.1 Registers Overview

The Gigabit Ethernet Controller registers materialize in PCI space. For more information on the conventions the following register summaries adopt, see Section 7.1, “Overview of Register Descriptions and Summaries” on page 183.

Note: The completion of any CSR access will be held off (maximum estimated delay 5ms) if initiated prior to completion of HW initialization (e.g.-EEPROM read). This behavior is transparent to software.

Table 37-21, Table 37-22, and Table 37-23 summarize the Gigabit Ethernet interface #0, #1, and #2 materializations from the PCI perspective.

Table 37-21. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	“CTRL: Device Control Register” on page 1438	00000A09h
0008h	000Bh	“STATUS: Device Status Register” on page 1441	0000XXXXh
0018h	001Bh	“CTRL_EXT: Extended Device Control Register” on page 1442	00000000h
00E0h	00E3h	“CTRL_AUX: Auxiliary Device Control Register” on page 1444	00000100h
0010h	0013h	“EEPROM_CTRL - EEPROM Control Register” on page 1446	00000X1Xh
0014h	0017h	“EEPROM_RR – EEPROM Read Register” on page 1448	XXXXXX00h
0028h	002Bh	“FCAL: Flow Control Address Low Register” on page 1449	00c28001h
002Ch	002Fh	“FCAH: Flow Control Address High Register” on page 1450	00000100h
0030h	0033h	“FCT: Flow Control Type Register” on page 1451	00008808h
0038h	003Bh	“VET: VLAN EtherType Register” on page 1452	00008100h
0170h	0173h	“FCTTV: Flow Control Transmit Timer Value Register” on page 1452	00000000h
1000h	1003h	“PBA: Packet Buffer Allocation Register” on page 1453	00100030h
00C0h	00C3h	“ICR0: Interrupt 0 Cause Read Register” on page 1454	00000000h
00C4h	00C7h	“ITR0: Interrupt 0 Throttling Register” on page 1457	00000000h
00C8h	00CBh	“ICS0: Interrupt 0 Cause Set Register” on page 1458	00000000h

Table 37-21. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h

Table 37-21. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h

Table 37-21. Bus M, Device 0, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5843h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh
05884h	5887h	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

Table 37-22. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"CTRL: Device Control Register" on page 1438	00000A09h
0008h	000Bh	"STATUS: Device Status Register" on page 1441	0000XXXXh
0018h	001Bh	"CTRL_EXT: Extended Device Control Register" on page 1442	00000000h
00E0h	00E3h	"CTRL_AUX: Auxiliary Device Control Register" on page 1444	00000100h
0010h	0013h	"EEPROM_CTRL - EEPROM Control Register" on page 1446	00000X1Xh
0014h	0017h	"EEPROM_RR – EEPROM Read Register" on page 1448	XXXXXX00h
0028h	002Bh	"FCAL: Flow Control Address Low Register" on page 1449	00c28001h
002Ch	002Fh	"FCAH: Flow Control Address High Register" on page 1450	00000100h
0030h	0033h	"FCT: Flow Control Type Register" on page 1451	00008808h
0038h	003Bh	"VET: VLAN EtherType Register" on page 1452	00008100h
0170h	0173h	"FCTTV: Flow Control Transmit Timer Value Register" on page 1452	00000000h
1000h	1003h	"PBA: Packet Buffer Allocation Register" on page 1453	00100030h
00C0h	00C3h	"ICR0: Interrupt 0 Cause Read Register" on page 1454	00000000h
00C4h	00C7h	"ITR0: Interrupt 0 Throttling Register" on page 1457	00000000h
00C8h	00CBh	"ICS0: Interrupt 0 Cause Set Register" on page 1458	00000000h
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h

Table 37-22. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h

Table 37-22. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5843h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh
05884h	5887h	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh

Table 37-22. Bus M, Device 1, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

Table 37-23. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 1 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
0000h	0003h	"CTRL: Device Control Register" on page 1438	00000A09h
0008h	000Bh	"STATUS: Device Status Register" on page 1441	0000XXXXh
0018h	001Bh	"CTRL_EXT: Extended Device Control Register" on page 1442	00000000h
00E0h	00E3h	"CTRL_AUX: Auxiliary Device Control Register" on page 1444	00000100h
0010h	0013h	"EEPROM_CTRL - EEPROM Control Register" on page 1446	00000X1Xh
0014h	0017h	"EEPROM_RR – EEPROM Read Register" on page 1448	XXXXXX00h
0028h	002Bh	"FCAL: Flow Control Address Low Register" on page 1449	00c28001h
002Ch	002Fh	"FCAH: Flow Control Address High Register" on page 1450	00000100h
0030h	0033h	"FCT: Flow Control Type Register" on page 1451	00008808h
0038h	003Bh	"VET: VLAN EtherType Register" on page 1452	00008100h
0170h	0173h	"FCTTV: Flow Control Transmit Timer Value Register" on page 1452	00000000h
1000h	1003h	"PBA: Packet Buffer Allocation Register" on page 1453	00100030h
00C0h	00C3h	"ICR0: Interrupt 0 Cause Read Register" on page 1454	00000000h
00C4h	00C7h	"ITR0: Interrupt 0 Throttling Register" on page 1457	00000000h
00C8h	00CBh	"ICS0: Interrupt 0 Cause Set Register" on page 1458	00000000h
00D0h	00D3h	"IMS0: Interrupt 0 Mask Set/Read Register" on page 1459	00000000h
00D8h	00DBh	"IMC0: Interrupt 0 Mask Clear Register" on page 1460	00000000h
08C0h	08C3h	"ICR1: Interrupt 1Cause Read Register" on page 1462	00000000h
08C8h	08CBh	"ICS1: Interrupt 0 Cause Set Register" on page 1464	00000000h
08D0h	08D3h	"IMS1: Interrupt 1 Mask Set/Read Register" on page 1466	00000000h
08D8h	08DBh	"IMC1: Interrupt 1 Mask Clear Register" on page 1467	00000000h
08E0h	08E3h	"ICR2: Error Interrupt Cause Read Register" on page 1469	00000000h
08E8h	08EBh	"ICS2: Error Interrupt Cause Set Register" on page 1471	00000000h
08F0h	08F3h	"IMS2: Error Interrupt Mask Set/Read Register" on page 1472	00000000h

Table 37-23. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 2 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
08F8h	08FBh	"IMC2: Error Interrupt Mask Clear Register" on page 1473	00000000h
0100h	0103h	"RCTL: Receive Control Register" on page 1474	00000000h
2160h	2163h	"FCRTL: Flow Control Receive Threshold Low Register" on page 1478	00000000h
2168h	216Bh	"FCRTH: Flow Control Receive Threshold High Register" on page 1479	00000000h
2800h	2803h	"RDBAL: Receive Descriptor Base Address Low Register" on page 1480	XXXXXXX0h
2804h	2807h	"RDBAH: Receive Descriptor Base Address High Register" on page 1480	XXXXXXXXh
2808h	280Bh	"RDLEN: Receive Descriptor Length Register" on page 1481	00000000h
2810h	2813h	"RDH: Receive Descriptor Head Register" on page 1481	00000000h
2818h	281Bh	"RDT: Receive Descriptor Tail Register" on page 1482	00000000h
2820h	2823h	"RDTR: RX Interrupt Delay Timer (Packet Timer) Register" on page 1483	00000000h
2828h	282Bh	"RXDCTL: Receive Descriptor Control Register" on page 1483	00010000h
282Ch	282Fh	"RADV: Receive Interrupt Absolute Delay Timer Register" on page 1485	00000000h
2C00h	2C03h	"RSRPD: Receive Small Packet Detect Interrupt Register" on page 1486	00000000h
5000h	5003h	"RXCSUM: Receive Checksum Control Register" on page 1487	00000000h
5200h at 4h	5203h at 4h	"MTA[0-127] – 128 Multicast Table Array Registers" on page 1488	XXXX_XXXXh
5400h at 8h	5403h at 8h	"RAL[0-15] - Receive Address Low Register" on page 1488	XXXXXXXXh
5404h at 8h	5407h at 8h	"RAH[0-15] - Receive Address High Register" on page 1489	000XXXXXh
5600h at 4h	5603h at 4h	"VFTA[0-127] - 128 VLAN Filter Table Array Registers" on page 1490	XXXXXXXXh
0400h	0403h	"TCTL: Transmit Control Register" on page 1491	00000008h
0410h	0413h	"TIPG: Transmit IPG Register" on page 1493	00602008h
0458h	045Bh	"AIT: Adaptive IFS Throttle Register" on page 1495	00000000h
3800h	3803h	"TDBAL: Transmit Descriptor Base Address Low Register" on page 1496	XXXXXXX0h
3804h	3807h	"TDBAH: Transmit Descriptor Base Address High Register" on page 1496	XXXXXXXXh
3808h	380Bh	"TDLEN: Transmit Descriptor Length Register" on page 1497	00000000h
3810h	3813h	"TDH: Transmit Descriptor Head Register" on page 1497	00000000h
3818h	381Bh	"TDT: Transmit Descriptor Tail Register" on page 1498	00000000h
3820h	3823h	"TIDV: Transmit Interrupt Delay Value Register" on page 1499	00000000h
3828h	382Bh	"TXDCTL: Transmit Descriptor Control Register" on page 1500	00000000h
382Ch	382Fh	"TADV: Transmit Absolute Interrupt Delay Value Register" on page 1502	00000000h
3830h	3833h	"TSPMT: TCP Segmentation Pad And Minimum Threshold Register" on page 1504	01000400h
4000h	4003h	"CRCERRS: CRC Error Count Register" on page 1505	00000000h
4004h	4007h	"ALGNERRC: Alignment Error Count Register" on page 1506	00000000h
400Ch	400Fh	"RXERRC: Receive Error Count Register" on page 1506	00000000h
4010h	4013h	"MPC: Missed Packet Count Register" on page 1507	00000000h
4014h	4017h	"SCC: Single Collision Count Register" on page 1507	0000h
4018h	401Bh	"ECOL: Excessive Collisions Count Register" on page 1508	00000000h
401Ch	401Fh	"MCC: Multiple Collision Count Register" on page 1508	00000000h
4020h	4023h	"LATECOL: Late Collisions Count Register" on page 1509	00000000h
4028h	402Bh	"COLC: Collision Count Register" on page 1509	00000000h

Table 37-23. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 3 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
4030h	4033h	"DC: Defer Count Register" on page 1510	00000000h
4034h	4037h	"TNCRS: Transmit with No CRS Count Register" on page 1510	00000000h
403Ch	403Fh	"CEXTERR: Carrier Extension Error Count Register" on page 1511	00000000h
4040h	4043h	"RLEC: Receive Length Error Count Register" on page 1511	00000000h
4048h	404Bh	"XONRXC: XON Received Count Register" on page 1512	00000000h
404Ch	404Fh	"XONTXC: XON Transmitted Count Register" on page 1512	00000000h
4050h	4053h	"XOFFRXC: XOFF Received Count Register" on page 1513	00000000h
4054h	4057h	"XOFFTXC: XOFF Transmitted Count Register" on page 1513	00000000h
4058h	405Bh	"FCRUC: FC Received Unsupported Count Register" on page 1514	00000000h
405Ch	405Fh	"PRC64: Good Packets Received Count (64 Bytes) Register" on page 1514	00000000h
4060h	4063h	"PRC127: Good Packets Received Count (65-127 Bytes) Register" on page 1515	00000000h
4064h	4067h	"PRC255: Good Packets Received Count (128-255 Bytes) Register" on page 1515	00000000h
4068h	406Bh	"PRC511 - Good Packets Received Count (256-511 Bytes) Register" on page 1516	00000000h
406Ch	406Fh	"PRC1023: Good Packets Received Count (512-1023 Bytes) Register" on page 1516	00000000h
4070h	4073h	"PRC1522: Good Packets Received Count (1024 to Max Bytes) Register" on page 1517	00000000h
4074h	4077h	"GPRC: Good Packets Received Count (Total) Register" on page 1518	00000000h
4078h	407Bh	"BPRC: Broadcast Packets Received Count Register" on page 1518	00000000h
407Ch	407Fh	"MPRC: Multicast Packets Received Count Register" on page 1519	00000000h
4080h	4083h	"GPTC: Good Packets Transmitted Count Register" on page 1519	00000000h
4088h	408Ah	"GORCL: Good Octets Received Count Low Register" on page 1520	00000000h
408Ch	408Fh	"GORCH: Good Octets Received Count High Register" on page 1521	00000000h
4090h	4093h	"GOTCL: Good Octets Transmitted Count Low Register" on page 1522	00000000h
4094h	4097h	"GOTCH: Good Octets Transmitted Count High Register" on page 1522	00000000h
40A0h	40A3h	"RNBC: Receive No Buffers Count Register" on page 1523	00000000h
40A4h	40A7h	"RUC: Receive Undersize Count Register" on page 1523	00000000h
40A8h	40ABh	"RFC: Receive Fragment Count Register" on page 1524	00000000h
40ACh	40AFh	"ROC: Receive Oversize Count Register" on page 1524	00000000h
40B0h	40B3h	"RJC: Receive Jabber Count Register" on page 1525	00000000h
40C0h	40C3h	"TORL: Total Octets Received Low Register" on page 1526	00000000h
40C4h	40C7h	"TORH: Total Octets Received High Register" on page 1526	00000000h
40C8h	40CFh	"TOTL: Total Octets Transmitted Low Register" on page 1527	00000000h
40CCh	40CFh	"TOTH: Total Octets Transmitted High Register" on page 1528	00000000h
40D0h	40D3h	"TPR: Total Packets Received Register" on page 1528	00000000h
40D4h	40D7h	"TPT: Total Packets Transmitted Register" on page 1529	00000000h
40D8h	40DBh	"PTC64 - Packets Transmitted Count (64 Bytes) Register" on page 1529	00000000h
40E0h	40E3h	"PTC255: Packets Transmitted Count (128-255 Bytes) Register" on page 1530	00000000h
40E4h	40E7h	"PTC511: Packets Transmitted Count (256-511 Bytes) Register" on page 1530	00000000h
40E8h	40EBh	"PTC1023: Packets Transmitted Count (512-1023 Bytes) Register" on page 1531	00000000h

Table 37-23. Bus M, Devices 2, Function 0: Summary of Gigabit Ethernet Interface Registers Mapped Through CSRBAR Memory BAR (Sheet 4 of 4)

Offset Start	Offset End	Register ID - Description	Default Value
40ECh	40EFh	"PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register" on page 1531	00000000h
40F0h	40F3h	"MPTC: Multicast Packets Transmitted Count Register" on page 1532	00000000h
40F4h	40F7h	"BPTC: Broadcast Packets Transmitted Count Register" on page 1532	00000000h
40F8h	40FBh	"TSCTC: TCP Segmentation Context Transmitted Count Register" on page 1533	00000000h
40FCh	40FFh	"TSCTFC: TCP Segmentation Context Transmit Fail Count Register" on page 1533	00000000h
5800h	5803h	"WUC - Wake Up Control Register (0x05800; RW)" on page 1534	00000000h
5808h	580Bh	"WUFC - Wake Up Filter Control Register (0x05808; RW)" on page 1535	00000000h
5810h	5813h	"WUS - Wake Up Status Register (0x05810; RW)" on page 1536	00000000h
5838h	583Bh	"IPAV - IP Address Valid Register (0x05838; RW)" on page 1537	00000000h
5840h at 8h	5607h at 8h	"IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers" on page 1538	XXXXXXXXh
5880h	5883h	"IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4" on page 1539	XXXXXXXXh
05884h	0588Fh	"IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8" on page 1539	XXXXXXXXh
5888h	588Bh	"IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12" on page 1540	XXXXXXXXh
588Ch	588Fh	"IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16" on page 1541	XXXXXXXXh
5F00h at 8h	5F03h at 8h	"FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)" on page 1542	00000000h
9000h at 8h	9003h at 8h	"FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)" on page 1543	0000000Xh
9800h at 8h	9803h at 8h	"FFVT[0-127]: Flexible Filter Value Table Registers" on page 1544	XXXXXXXXh
0510h	0513h	"INTBUS_ERR_STAT - Internal Bus Error Status Register" on page 1544	00000000h
0900h	0903h	"MEM_TST - Memory Error Test Register" on page 1546	00000000h
0904h	0907h	"MEM_STS - Memory Error Status Register" on page 1547	007F0000h

37.6.1.1 Memory-Mapped Access to Internal Registers and Memories

The internal registers and memories may be accessed as direct memory-mapped offsets from the base address register. See Section 37.6, "GbE Controller Register Summary" for the appropriate offset for each specific internal register.

37.6.1.2 I/O-Mapped Access to Internal Registers and Memories

To support pre-boot operation (prior to the allocation of physical memory base addresses), all internal registers and memories can be accessed using I/O operations. I/O accesses are supported only if an I/O Base Address is allocated and mapped, the BAR contains a valid (non-zero value), and I/O address decoding is enabled in the PCI/PCIX configuration.

When an I/O BAR is mapped, the I/O address range allocated opens a 32-byte "window" in the system I/O address map. Within this window, two I/O addressable register are implemented: IOADDR and IODATA. The IOADDR register is used to specify a reference to an internal register or memory and then the IODATA register is used as a "window" to the register or memory address specified by IOADDR:

Table 37-24. I/O Mapped Registers

Offset	Abbreviation	Name	R/W	Size
0x00	IOADDR	Internal Register or Internal Memory Location Address. 0x00000-0x1FFFF – Internal Registers and Memories 0x20000-0xFFFFF – Undefined	RW	4 bytes
0x04	IODATA	Data field for reads or writes to the Internal Register or Internal Memory Location as identified by the current value in IOADDR. All 32 bits of this register are read/write-able.	RW	4 bytes
0x08 – 0x3F	Reserved	Reserved	RW	None

37.6.1.2.1 IOADDR (I/O offset 0x00)

The IOADDR register must always be written as a DWORD access (e.g. the PCI_CBE_N[3:0] byte enables must all be enabled). Writes that are less than 32 bits will be ignored. Reads of any size will return a DWORD of data. However, the chipset or CPU may only return a subset of that DWORD.

For software programmers, the IN and OUT instructions must be used to cause I/O cycles to be used on the PCI bus. Because writes must be to a 32-bit quantity, the source register of the OUT instruction must be EAX (the only 32-bit register supported by the OUT command). For reads, the IN instruction can have any size target register, but it is recommended that the 32-bit EAX register be used.

Because only a particular range is addressable, the upper bits of this register are hard coded to zero. Bits 31 through 20 are not write-able and always read back as 0b.

At hardware reset (LAN_PWR_GOOD) or PCI Reset, this register value resets to 0x00000000. Once written, the value is retained until the next write or reset.

37.6.1.2.2 IODATA (I/O offset 0x04)

The IODATA register must always be written as a DWORD access when the IOADDR register contains a value for the Internal Register and Memories (e.g. 0x00000-0x1FFFC). In this case, writes that are less than 32 bits will be ignored.

Writes and reads to IODATA when the IOADDR register value is in an undefined range (0x20000-0x7FFFC) should not be performed. Results are indeterministic.

Note: There are no special software timing requirements on accesses to IOADDR or IODATA. All accesses will be immediate except when data is not readily available or acceptable. In this case, the MAC will delay the results through normal bus methods.

Note: Because a register/memory read or write takes two I/O cycles to complete, software must provide a guarantee that the two I/O cycles occur as an atomic operation. Otherwise, results can be non-deterministic from the software viewpoint.

37.6.1.2.3 Undefined I/O offsets

I/O offsets 0x08 through 0x3F are considered to reserved offsets with the I/O window. Writes within this address region may cause unpredictable behavior. Reads within this address region may return indeterminate values.

37.6.1.3 Register Conventions

All registers in the GbE are defined to be 32 bits and must be accessed with a DWORD transaction.

- **Reserved bit positions —** Some registers contain certain bits that are marked as “reserved”. These bits should never be written with anything other than their initial value by software (indicated in their individual descriptions). Reads from registers containing reserved bits may return indeterminate values in the reserved bit-positions unless read values are explicitly stated. When read, these reserved bits should be ignored by software.
- **Reserved and/or undefined addresses —** Any register address not explicitly declared in this specification should be considered to be reserved, and should not be written. Writing to reserved or undefined register addresses may cause indeterminate behavior. Reads from reserved or undefined configuration register addresses may return indeterminate values unless read values are explicitly stated for specific addresses.
- **Initial values —** Most registers define the initial hardware values prior to being programmed. In some cases, hardware initial values are undefined and will be listed as such via the text “undefined”, “unknown”, or “X”. Some such configuration values may need to be set by software in order for proper operation to occur; this need is dependent on the function of the bit. Other registers may cite a hardware default which is overridden by a higher-precedence operation. Operations which may supersede hardware defaults may include completion of a hardware operation (such as hardware auto-negotiation), or writing of a different register whose value is then reflected in another bit.

For all registers, partial reads and writes may cause indeterminate behavior.

37.6.2 General Registers: Detailed Descriptions

37.6.2.1 CTRL – Device Control Register

This register, as well as the Extended Device Control Register (CTRL_EXT), controls the major operational modes for the device. While software write to this register to control device settings, several bits (such as FD and SPEED) may be overridden depending on other bit settings and the resultant link configuration determined by the Auto-Negotiation resolution with the PHY. See "Physical Layer Auto-Negotiation & Link Setup Features" on page 1394 for a detailed explanation on the link configuration process.

Table 37-25. CTRL: Device Control Register (Sheet 1 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0000h **Offset End:** 0003h	
Size: 32 bits	**Default:** 00000A09h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31	Rsvd	Reserved		0h	RV
30	VME	VLAN Mode Enable. 0 = VLAN Mode Disabled. 1 = VLAN Mode Enabled. All packets transmitted have an 802.1q header added to the packet. The contents of the header come from the transmit descriptor and from the VLAN type register. On receive, VLAN information is stripped from 802.1q packets. See "802.1q VLAN Support" on page 1400 for more details.		0h	RW
29	Rsvd	Reserved		0h	RV
28	TFCE	Transmit Flow Control Enable. 0 = **Transmit Flow Control** Disabled. 1 = **Transmit Flow Control** Enabled. Flow control packets (XON & XOFF frames) will be transmitted based on receiver fullness. If Auto-Negotiation is enabled, this bit is set to the negotiated duplex value. See "Physical Layer Auto-Negotiation & Link Setup Features" on page 1394 for more information about Auto-Negotiation.		0h	RW
27	RFCE	Receive Flow Control Enable. 0 = **Receive Flow Control** Disabled. 1 = **Receive Flow Control** Enabled. Indicates the device will respond to the reception of flow control packets. Reception of flow control packets requires the correct loading of the FCAH/FCAL & FCT registers. If Auto-Negotiation is enabled, this bit is set to the negotiated duplex value. See "Physical Layer Auto-Negotiation & Link Setup Features" on page 1394 for more information about Auto-Negotiation.		0h	RW

Table 37-25.CTRL: Device Control Register (Sheet 2 of 3)

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 0000h **Offset End:** 0003h	
Size: 32 bits	**Default:** 00000A09h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
26	RST	Device Reset, also referred to as a “Soft Reset”. Normally 0, writing 1 initiates the reset. This bit is self clearing. CTRL.RST may be used to globally reset the entire GbE hardware. This register is provided primarily as a last-ditch software mechanism to recover from an indeterminate or suspected hung hardware state. Most registers (receive, transmit, interrupt, statistics, etc.), and state machines will be set to their power-on reset values, approximating the state following a power-on or Unit Reset. However, the Packet Buffer Allocation Register (PBA) retains its value through a global reset. ***Note:*** Software must first disable both transmit & receive operation using the TCTL.EN and RCTL.EN register bits before asserting CTRL.RST. To ensure that the global device reset has fully completed and that the controller will respond to subsequent accesses, software must wait a minimum of 5 milliseconds after setting CTRL.RST before attempting to check if the bit has cleared or to access any other GbE device register.			0h	RW
25 :21	Rsvd	Reserved			0h	RV
20	ADVD3WUC	D3Cold WakeUp Capability Advertisement Enable. When set, D3Cold wakeup capability may be advertised based on whether the AUX_PWR pin advertises presence of auxiliary power (see section 2.13.3 for details). When 0, D3Cold wakeup capability will not be advertised even if AUX_PWR presence is indicated. Formerly used as SDP2 pin data value, initial value is EEPROM-configurable *Note that this bit is loaded from the EEPROM, if present			0h	RW
19 : 13	Rsvd	Reserved			0h	RV
12	FRCDPLX	Force Duplex. 0 = Mode is Full-Duplex, regardless of the FD setting. 1 = CTRL.FD bit sets duplex mode.			0h	RW
11	FRCSPD	Force Speed. 0 = Default of 1Gbps is used to set the MAC speed. See “Physical Layer Auto-Negotiation & Link Setup Features” on page 1394 for more details. 1 = CTRL.SPEED bits set the MAC speed. ***Note:*** This bit is superseded by the CTRL_EXT.SPD_BYPS bit which has a similar function. ***Note:*** *Note that this bit is loaded from the EEPROM, if present			1	RW
10	Rsvd	Reserved			0h	RV

Table 37-25. CTRL: Device Control Register (Sheet 3 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0000h **Offset End:** 0003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0000h **Offset End:** 0003h	
Size: 32 bits	**Default:** 00000A09h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
09 : 08	SPEED	Speed selection. These bits are written by software (assuming, after reading the PHY registers through the MDIO interface) to set the MAC speed configuration. See "Physical Layer Auto-Negotiation & Link Setup Features" on page 1394 for details. • 00 => 10 Mbps • 01 => 100 Mbps • 10 => 1000 Mbps • 11 => reserved ***Note:*** These bits affect the MAC speed setting only if CTRL_EXT.SPD_BYPS or CTRL.FRCSPD are used.		10b	RW
07	Reserved	Reserved		0h	RV
06	SLU	Set Link Up. SLU must be set to '1' to enable the MAC. This bit may also be initialized by the APME bit in the EEPROM Initialization Control Word3, if an EEPROM is used.		0h	RW
05	Rsvd	Reserved. Must be set to 0.		0h	RW
04	Rsvd	Reserved		0h	RV
03	Rsvd	Reserved		1	RW
02	Rsvd	Reserved		0h	RV
01	Rsvd	Reserved. Must write '0' to this bit. 1 =		0h	RW
00	FD	**Full Duplex.** Controls the MAC duplex setting. 0 = Half Duplex 1 = Full Duplex In half-duplex mode, EP80579's GbE transmits carrier extended packets and can receive both carrier extended packets, and packets transmitted with bursting. *Note that this bit is loaded from the EEPROM, if present		1	RW

37.6.2.2 STATUS – Device Status Register

This register reports the status of the major operational modes for the device. This is a read only register.

Table 37-26. STATUS: Device Status Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0008h **Offset End:** 000Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0008h **Offset End:** 000Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0008h **Offset End:** 000Bh	
Size: 32 bits	**Default:** 0000XXXXh			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 10	Rsvd	Reserved		0h	RV
09 : 08	Reserved	Reserved		X	RO
07 : 06	SPEED	**Link Speed Setting:** Reflects speed setting of the MAC. In GMII/MII mode, these bits reflect the software CTRL.SPEED setting • 00 => 10 Mbps • 01 => 100 Mbps • 10 => 1000 Mbps • 11 => 1000 Mbps		X	RO
05	LINKMODE	Mode. Based on CTRL_EXT. LINK_MODE. 0 = MAC is operating in GMII/MII mode 1 = Reserved		X	RO
04	TXOFF	Transmission Off. This bit indicates the state of the transmit function when symmetrical flow control has been enabled and negotiated with the link partner. 0 = Symmetrical flow control is disabled, or transmission is not paused. 1 = Symmetrical flow control is enabled, and the transmit function is paused due to the reception of an XOFF frame. It is cleared upon expiration of the pause timer or the receipt of an XON frame.		X	RO
03 : 02	Rsvd	Reserved		0h	RV
01	RSVD	Reserved		X	RO
00	FD	**Full Duplex.** This bit reflects the MAC duplex configuration. Normally, the duplex setting for the link, as it should reflect the duplex configuration negotiated between the PHY and link partner (copper link) or MAC and link partner (fiber link). 0 = Half Duplex mode 1 = Full Duplex mode		X	RO

37.6.2.3 CTRL_EXT – Extended Device Control Register

This register provides extended control of device functionality beyond that provided by the Device Control Register (CTRL).

Table 37-27. CTRL_EXT: Extended Device Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0018h **Offset End:** 001Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0018h **Offset End:** 001Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0018h **Offset End:** 001Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 25	Rsvd	Reserved		0h	RV
24	RMII_RX_MODE	RMII gasket receive mode select: 0 = For proper 100mbps receive operation, after assertion of the RMII CRS_DV signal on GBEn_RXCTL, the RMII gasket requires that a minimum of two di-bits of '00' appear on GBEn_RXDATA[1:0] before the preamble appears. 1 = For proper 100mbps receive operation, the RMII gasket requires that CRS_DV be asserted on GBEn_RXCTL synchronously with GBE_REFCLK_RMII and on the same cycle in which the first di-bit of the preamble appears on GBEn_RXDATA[1:0]. 0 is the default value of this bit and makes the RMII gasket compatible with RMII PHYs that assert CRS_DV as soon as the receive medium is non-idle, and subsequently drive '00' on RXD[1:0] until proper receive signal decoding has been achieved (per the RMII Specification, Revision 1.2). Setting this bit to a 1 makes the gasket compatible with RMII PHYs that assert CRS_DV simultaneously with the start of the preamble driven on RXD[1:0]. While this CRS_DV signalling mode does not scrictly conform to the RMII specification, it is provided to allow compatibility with PHY devices that use this alternate method of asserting CRS_DV at the start of the packet. This bit must be set to the proper state that corresponds to the CRS_DV behavior of the attached RMII PHY, otherwise 100mbps packets cannot be properly received by the GbE. This bit does not affect transmit operations.		0h	RW
23 : 22	LINK_MODE	**Link Mode.** This controls which interface is used to talk to the link. • 00 => GMII/MII mode • 01 => reserved • 10 => reserved • 11 => reserved • *Note that this bit is loaded from the EEPROM, if present		0h	RW
21 : 16	Rsvd	Reserved		0h	RV

Table 37-27.CTRL_EXT: Extended Device Control Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0018h **Offset End:** 001Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0018h **Offset End:** 001Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0018h **Offset End:** 001Bh	
Size: 32 bit	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15	SPD_BYPS	Speed Select Bypass. 0 = Normal speed detection mechanisms are used to determine the speed of the MAC. 1 = All speed detection mechanisms are bypassed and the MAC is immediately set to the setting of CTRL.SPEED. ***Note:*** CTRL_EXT.SPD_BYPS performs a function similar to CTRL.FRCSPD in that the device's speed settings are determined by the value software writes to the CTRL.SPEED bits. However, when using CTRL_EXT.SPD_BYPS the CTRL.SPEED setting takes effect immediately, when using CTRL.FRCSPD the CTRL.SPEED setting waits until after the device's clock switching circuitry performs the change.		0h	RW
14	Rsvd	Reserved		0h	RV
13	EE_RST	EEPROM Reset Initiates a “reset-like” event to the EEPROM function. This causes the EEPROM to be read as if a UNIT_RESET had occurred. All device functions should be disabled prior to setting this bit. This bit is self-clearing. NOTE: this will not cause the controller to detect the EEPROM		0h	RW
12	Rsvd	Reserved		0h	RV
11 : 00	Rsvd	Reserved		0h	RV

37.6.2.4 CTRL_AUX – Auxiliary Device Control/Status Register

This register provides extended control of device functionality beyond that provided by the Device Control Register (CTRL) and Extended Device Control Register

Table 37-28. CTRL_AUX: Auxiliary Device Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
Size: 32 bits	**Default:** 00000100h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :18	RSVD	Reserved		0h	RO
17	RMII_LOG_FIX	Enable logic change to fix RMII 100mbps TX dropped packet data. To enable this mode of operation, set this bit to a '1'. When enabled, the fix modifies the legacy new-packet signalling logic in the transmit path to prevent the first 8 bytes of packet data from being dropped when operating in RMII mode and a line speed of 100mbps.		0h	RW
16	RMII_FREQ_FIX	Disable DMA frequency change to fix RMII 100mbps TX dropped packet data. This is the default mode of operation. To disable this mode of operation, set this bit to a '1'. This must be disabled if FIX2 is enabled. When enabled, sets the DMA clock frequency to 50MHz when operating in RMII mode. This produces a favorable frequency ratio between DMA and MAC clocks that prevents the first 8 bytes of transmit packet data from being dropped when operating in RMII mode and a line speed of 100mbps.		0h	RW
15 :12	RSVD	Reserved		0h	RO
11 :10	END_SEL	Selects whether the descriptor or packet data is controlled by endianness configuration. 00 - descriptor and packet transfers use CTRL_AUX.ENDIANESS 01 - descriptor uses CTL_AUX.ENDIANESS, packet uses default 10 - descriptor uses default, packet uses CTRL_AUX.ENDIANESS 11 - all transfers use CTRL_AUX.ENDIANESS		0h	RW

Table 37-28.CTRL_AUX: Auxiliary Device Control Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00E0h **Offset End:** 00E3h	
Size: 32 bits	**Default:** 00000100h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
09 : 08	ENDIANESS	Endianness: These bits control the endianness of the data in memory. These settings apply to all internal bus transactions, including packet data and descriptors '00' - LW Little--Endian, Byte Big-Endian '01' - LW Little-Endian, Byte Little-Endian (default) '10' - LW Big-Endian, Byte Big-Endian '11' - LW Big-Endian, Byte Little-Endian Refer to Section 37.5.14, "Endianness" for further details.		01h	RW
07 : 01	RSVD	Reserved		0h	RO
00	RGMII_RMII	RGMII/RMII Translation Gasket Select • '0' - RGMII • '1' - RMII		0h	RW

37.6.2.5 EEPROM_CTRL – EEPROM Control Register

Table 37-29. EEPROM_CTRL - EEPROM Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0010h **Offset End:** 0013h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0010h **Offset End:** 0013h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0010h **Offset End:** 0013h	
Size: 32 bits	**Default:** 00000X1Xh			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 10	RSVD	Reserved		0h	RO
09	EE_SIZE	**EEPROM Size**. 0 Reserved 1 4096-bit (256 word) NM93C66 compatible EEPROM If an EEPROM is present, this bit indicates its size, based on acknowledges seen during EEPROM scans of different addresses. This bit is read-only. NOTE: this bit will not be updated as a result of anything but a power up reset.		Xh	RO
08	EE_PRES	EEPROM Present This bit attempts to indicate if an EEPROM is present by monitoring the EE_DO input for a active-low “acknowledge” by the serial EEPROM during initial EEPROM scan. If no EEPROM is present, the EE_DO line will remain pulled-high and thus no acknowledge will be seen. 1=EEPROM present; 0=no EEPROM. NOTE: this bit will not be set except as a result of EEPROM detection during power up reset.		Xh	RO
07	EE_GNT	Grant EEPROM Access When this bit is 1 the software can access the EEPROM using the SK, CS, DI, and DO bits.		0h	RO
06	EE_REQ	Request EEPROM Access The software must write a 1 to this bit to get direct EEPROM access. It has access when EE_GNT is 1. When the software completes the access it must write a 0.		0h	RW
05 : 04	RSVD	Reserved		01h	RO
03	EE_DO	**Data Output Bit from the EEPROM**. The EE_DO input signal is mapped directly to this bit in the register and contains the EEPROM data output. This bit is read-only from the software perspective – writes to this bit have no effect.		X	RO

Table 37-29. EEPROM_CTRL - EEPROM Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 0010h **Offset End:** 0013h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 0010h **Offset End:** 0013h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 0010h **Offset End:** 0013h	
Size: 32 bits	**Default:** 00000X1Xh				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
02	EE_DI	**Data Input to the EEPROM**. When EE_GNT is 1, the EE_DI output signal is mapped directly to this bit. Software provides data input to the EEPROM via writes to this bit.			0h	RW
01	EE_CS	**Chip Select Input to the EEPROM**. When EE_GNT is 1, the EE_CS output signal is mapped to the chip select of the EEPROM device. Software enables the EEPROM by writing a 1 to this bit.			0h	RW
00	EE_SK	**Clock Input to the EEPROM**. When EE_GNT is 1, the EE_SK output signal is mapped to this bit and provides the serial clock input to the EEPROM. Software clocks the EEPROM via toggling this bit with successive writes.			0h	RW

This register provides software direct access to the EEPROM. Software can control the EEPROM by successive writes to this register. Data & address information is clocked into the EEPROM by software toggling the EESK bit (2) of this register with EECS set to 1. Data output from the EEPROM is latched into bit 3 of this register via the internal 62.5MHz clock and may be accessed by software via reads of this register.

37.6.2.6 EEPROM_RR – EEPROM Read Register

Table 37-30. EEPROM_RR – EEPROM Read Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0014h **Offset End:** 0017h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0014h **Offset End:** 0017h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0014h **Offset End:** 0017h	
Size: 32 bits	**Default:** XXXXXX00h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 16	DATA	Read Data Data returned from the EEPROM read.		X	RO
15 : 08	ADDR	Read Address This field is written by software along with *Start Read* to indicate the word to read.		X	RW
07 : 05	RSVD	**Reserved** Reads as 0		0h	RV
04	DONE	Read Done Set to 1 when the EEPROM read completes. Set to 0 when the EEPROM read is in progress. Writes by software are ignored.		0h	RO
03 : 01	RSVD	**Reserved** Reads as 0		0h	RV
00	START	Start Read Writing a 1 to this bit causes the EEPROM to read a (16-bit) word at the address stored in the EE_ADDR field, storing the result in the EE_DATA field. This bit is self-clearing		0h	RW

This registers is used by software to read individual words in the EEPROM. To read a word, software writes the address to the Read Address field and simultaneously writes a 1 to the Start Read field. The GbE will read the word from the EEPROM and place it in the Read Data field, setting the Read Done filed to 1. Software can poll this register, looking for a 1 in the Read Done filed, and then using the value in the Read Data field.

When this register is used to read a word from the EEPROM, that word is not written to any of the MAC's internal registers even if it is normally a hardware accessed word.

Note: If SW has requested direct pin control of the EEPROM using the EEC register, an access through the EERD register mechanism may stall until the EEC control has been released. SW should ensure that EEC.EE_REQ=0 and that EEC.EE_GNT=0 as well before attempting to utilize EERD to access the EEPROM.

37.6.2.7 FCAL – Flow Control Address Low Register

Flow control packets are defined by 802.3X to be either a unique multicast address or the station address with the EtherType field indicating PAUSE. The FCAH and FCAL registers provide the value hardware compares incoming packets against to determine that it should PAUSE its output. This register contains the lower bits of the internal 48 bit Flow Control Ethernet address.

Note: Any packet matching the contents of {FCAH, FCAL, FCT} when CTRL.RFCE is set will be acted on by the EP80579's GbE. Whether flow control packets are passed to the software depends on the state of the RCTL.DPF bit and whether the packet matches any of the normal filters.

Note: At the time of the original implementation, the flow control multicast address was not defined and thus hardware provided programmability. Since then, the final release of the 802.3x standard has reserved the following multicast address for MAC Control Frames: 01-80-C2-00-00-01.

Note: This register MUST be written by software with the appropriate value for the MAC to behave properly.

Table 37-31. FCAL: Flow Control Address Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0028h **Offset End:** 002Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0028h **Offset End:** 002Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0028h **Offset End:** 002Bh	
Size: 32 bits	**Default:** 00c28001h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	FCAL	This register must be programmed with 0x00C2_8001.		00c28001h	RW

37.6.2.8 FCAH – Flow Control Address High Register

Flow control packets are defined by 802.3X to be either a unique multicast address or the station address with the EtherType field indicating PAUSE. The FCAH and FCAL registers provide the value hardware compares incoming packets against to determine that it should PAUSE its output. This register contains the upper bits of the internal 48 bit Flow Control Ethernet address.

Note: Any packet matching the contents of {FCAH, FCAL, FCT} when CTRL.RFCE is set will be acted on by the EP80579's GbE. Whether flow control packets are passed to the software depends on the state of the RCTL.DPF bit and whether the packet matches any of the normal filters.

Note: At the time of the original implementation, the flow control multicast address was not defined and thus hardware provided programmability. Since then, the final release of the 802.3x standard has reserved the following multicast address for MAC Control Frames: 01-80-C2-00-00-01.

Note: This register MUST be written by software with the appropriate value for the MAC to behave properly.

Table 37-32. FCAH: Flow Control Address High Register

Description:	This register must be programmed with 0x00000100.				
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:0:0		**Offset Start:** 002Ch **Offset End:** 002Fh	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:1:0		**Offset Start:** 002Ch **Offset End:** 002Fh	
View: PCI 3	**BAR:** CSRBAR	**Bus:Device:Function:** M:2:0		**Offset Start:** 002Ch **Offset End:** 002Fh	
Size: 32 bits	**Default:** 00000100h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description		Sticky	Bit Reset Value	Bit Access
31 16	RSVD	Reserved			0	RV
15 : 00	FCAH	This register must be programmed with 0x00_00_01_00.			0100h	RW

37.6.2.9 FCT – Flow Control Type Register

Flow control packets are defined by 802.3X to be either a unique multicast address or the station address with the EtherType field indicating PAUSE. The FCAH and FCAL registers provide the value hardware compares incoming packets against to determine that it should PAUSE its output. This register contains the type field hardware matches against to recognize a flow control packet.

Note: Any packet matching the contents of {FCAH, FCAL, FCT} when CTRL.RFCE is set will be acted on by the EP80579's GbE. Whether flow control packets are passed to the software depends on the state of the RCTL.DPF bit and whether the packet matches any of the normal filters.

Note: At the time of the original implementation, the flow control type field was not defined and thus hardware provided programmability. Since then, the final release of the 802.3x standard has specified the type/length value for MAC Control Frames as 88-08.

Table 37-33. FCT: Flow Control Type Register

Description:	This register must be programmed with 0x00_00_88_0					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 0030h **Offset End:** 0033h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 0030h **Offset End:** 0033h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 0030h **Offset End:** 0033h	
Size: 32 bits	**Default:** 00008808h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 16	RSVD	Reserved			0000h	RV
15 : 00	FCT	This register must be programmed with 0x00_00_88_08.			8808h	RW

37.6.2.10 VET – VLAN EtherType Register

This register contains the type field hardware matches against to recognize an 802.1Q (VLAN) Ethernet packet.

Table 37-34. VET: VLAN EtherType Register

Description:	To be compliant with the 802.3ac standard, this register must be programmed with the value 0x00_00_81_00				
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0038h **Offset End:** 003Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0038h **Offset End:** 003Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0038h **Offset End:** 003Bh	
Size: 32 bits	**Default:** 00008100h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 16	RSVD	Reserved		0	RV
15 : 00	VET	To be compliant with the 802.3ac standard, this register must be programmed with the value 0x00_00_81_00.		8100h	RW

37.6.2.11 FCTTV – Flow Control Transmit Timer Value Register

The register holds a 16-bit value in the TTV field that is inserted into a transmitted frame (either XOFF frames or any PAUSE frame value in any software transmitted packets). It counts in units of slot time, usually 64B. If software wishes to send an XON frame, it must set TTV to 0 prior to initiating the PAUSE frame.

Note: The EP80579's GbE uses a fixed slot time value of 64B times

Table 37-35. FCTTV: Flow Control Transmit Timer Value Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0170h **Offset End:** 0173h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0170h **Offset End:** 0173h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0170h **Offset End:** 0173h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	TTV	**Transmit Timer Value** to be included in XOFF frame.		0h	RW

37.6.2.12 PBA – Packet Buffer Allocation Register

This register sets the GbE hardware receive and transmit storage allocation ratio. The Tx allocation is calculated from the programmed Rx allocation, assuming 64KB of total Packet Buffer memory.

Note: Programming this register does not automatically re-load or initialize internal packet-buffer RAM pointers. The software must reset both transmit and receive operation (using the global device reset CTRL.RST bit) after changing this register in order for it to take effect. The PBA register itself will not be reset by assertion of the global reset, but will only be reset upon initial hardware power-on.

Table 37-36. PBA: Packet Buffer Allocation Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 1000h **Offset End:** 1003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 1000h **Offset End:** 1003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 1000h **Offset End:** 1003h	
Size: 32 bits	**Default:** 00100030h			**Power Well:**	GbE0: Core Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 :22	RSVD	Reserved		0h	RO
21 : 16	TXA	**Transmit Packet Buffer Allocation in K bytes.** PBA.TXA is read only and calculated based on PBA.RXA. 0010h =>16KB		0010h	RO
15 :6	RSVD	Reserved		0h	RO
5 : 00	RXA	**Receive Packet Buffer Allocation in K bytes.** PBA.RXA legal values must be 8K aligned. Valid values are (decimal) 8, 16, 24, 32, 40, 48, 56. 0030h => 48KBh		0030h	RW

37.6.3 Interrupt Registers: Detailed Descriptions

37.6.3.1 ICR0 – Interrupt 0 Cause Read Register

This register contains all interrupt conditions for the GbE. Whenever an interrupt causing event occurs, the corresponding interrupt bit is set in this register. See "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 for additional details.

Note: All register bits clear on read. Thus, reading this register implicitly acknowledges all pending interrupt events. Writing a 1 to any bit in the register will also clear that bit. Writing a 0 to any bit will have no effect.

Table 37-37. ICR0: Interrupt 0 Cause Read Register (Sheet 1 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 29	RSVD	Reserved		0h	RV
28	ERR_INTBUS	**Internal Bus Error.** This bit indicates that an error occurred during either a Target or Host transaction on the bus. Refer to Section 37.5.12, "Error Handling" for complete details. The details of this error are reported in the INTBUS_ERR_STAT register.		0h	RCWC
27	ERR_STAT	**Statistic Register ECC Error.** The Statistic Registers are implemented using a memory that uses a single-bit correct/multi-bit detect ECC parity algorithm to protect it. This bit indicates that a multi-bit error has occurred on a read from that memory. No indication of a single-bit error correction will be given by hardware. ***Note:*** *If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)*		0h	RCWC
26	ERR_MCFSPF	**This bit indicates that either a Multicast Filter Parity Error, Special Packet Filter Parity Error or a Flex Filter Parity Error occurred.** These filters use parity protected SRAMs for data buffers. This bit indicates that a parity error has occurred on a read from either of these data buffers. This error is considered non-fatal and will clear after a read of the MEM_ERR_STAT register.		0h	RCWC
25 : 24	RSVD	Reserved		0h	RV

Table 37-37. ICR0: Interrupt 0 Cause Read Register (Sheet 2 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00C0h **Offset End:** 00C3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
23	ERR_PB	**DMA Packet Buffer 2-bit ECC Error.** The 64KB DMA Packet Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for data. This bit indicates that a multi-bit error has occurred on a read from that SRAM. No indication of a single-bit error correction will be given by hardware. ***Note:*** *If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)*		0h	RCWC
22	RSVD	Reserved		0h	RV
21	ERR_TXDS	DMA Transmit Descriptor 2-bit ECC Error. The DMA Transmit Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)		0h	RCWC
20	ERR_RXDS	DMA Receive Descriptor 2-bit ECC Error. The DMA Receive Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)		0h	RCWC
19 : 17	RSVD	Reserved		0h	RV
16	SRPD	**Small Receive Packet Detected.** Indicates that a packet of size RSRPD.SIZE register has been detected and transferred to host memory. The interrupt is only asserted if RSRPD.SIZE register has a non-zero value		0h	RCWC
15	TXD_LOW	Transmit Descriptor Low Threshold hit. Indicates that the descriptor ring has reached the threshold specified in "TXDCTL – Transmit Descriptor Control Register" on page 1500.		0h	RCWC
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	**Receiver Timer Interrupt.** Set when the timers expire, see "Receive Interrupts" on page 1360 for details.		0h	RCWC
06	RXO	**Receiver Overrun.** Set on receive data FIFO overrun. Could be caused either because there are no available buffers or because Internal Bus receive bandwidth is inadequate.		0h	RCWC

Table 37-37. ICR0: Interrupt 0 Cause Read Register (Sheet 3 of 3)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 00C0h Offset End: 00C3h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 00C0h Offset End: 00C3h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 00C0h Offset End: 00C3h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
05	Rsvd	Reserved		0h	RV
04	RXDMT0	**Receive Descriptor Minimum Threshold Hit.** Indicates that the minimum number of receive descriptors are available and software should load more receive descriptors.		0h	RCWC
03	Rsvd	**Reserved**		0h	RCWC
02	Rsvd	**Reserved**		0h	RV
01	TXQE	**Transmit Queue Empty.** Set when the last descriptor block for a transmit queue has been used.		0h	RCWC
00	TXDW	**Transmit Descriptor Written Back.** Set when hardware processes a descriptor with its RS bit set. If using delayed interrupts (TDESC.IDE is set in the Transmit Descriptor CMD), the interrupt is delayed until after one of the delayed-timers (TIDV or TADV) expires.		0h	RCWC

37.6.3.2 ITR0 – Interrupt 0 Throttling Register

Software can use this register to pace (or balance) the delivery of the functional interrupt signal to the host CPU. This register provides a guaranteed inter-interrupt delay between interrupts asserted by the network controller, regardless of network traffic conditions.

Table 37-38. ITR0: Interrupt 0 Throttling Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 00C4h **Offset End:** 00C7h
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 00C4h **Offset End:** 00C7h
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 00C4h **Offset End:** 00C7h
Size: 32 bits	**Default:** 00000000h				**Power Well:** GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	MIII	**Minimum Inter-interrupt Interval.** • In RGMII mode, the interval is specified in 256ns increments. • In RMII mode, the interval is specified in 320ns increments • Zero disables interrupt throttling logic (The following example applies to RGMII mode) To independently validate configuration settings, software can use the following formula to convert the inter-interrupt interval value to the common 'interrupts/sec' performance metric: interrupts/sec = (256 x 10^{-9} sec x inter-interrupt interval)$^{-1}$ Inversely, inter-interrupt interval value can be calculated as: inter-interrupt interval = (256 x 10^{-9} sec x interrupts/sec)$^{-1}$ For example, if the interval is programmed to 500d, the network controller guarantees the CPU will not be interrupted by the network controller for 128 usec from the last interrupt. The maximum observable interrupt rate from the adapter should never exceed 7813 interrupts/sec. The optimal performance setting for this register is system/configuration specific. A initial suggested range is 651-5580 (28Bh - 15CCh), or, more generally, between 700 and 6000 interrupts per second.		0h	RW

37.6.3.3 ICS0 – Interrupt 0 Cause Set Register

Software uses this register to set an interrupt condition. Assuming the interrupt mask is set, any bit written with a 1 triggers the corresponding interrupt, see "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 and "ICR0 – Interrupt 0 Cause Read Register" on page 1454.

Table 37-39. ICS0: Interrupt 0 Cause Set Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00C8h **Offset End:** 00CBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00C8h **Offset End:** 00CBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00C8h **Offset End:** 00CBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Triggers Internal Bus Error		0h	RW
27	ERR_STAT	Triggers Statistic Register ECC Error		0h	RW
26	ERR_MCFSPF	Triggers Special Packet Filter Parity Error		0h	RW
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Triggers DMA Packet Buffer ECC Error		0h	RW
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Triggers DMA Transmit Descriptor Buffer ECC Error		0h	RW
20	ERR_RXDS	Triggers DMA Receive Descriptor Buffer ECC Error		0h	RW
19 : 17	Rsvd	Reserved		0h	RV
16	SRPD	Triggers Small Receive Packet Detected and Transferred		0h	RW
15	TXD_LOW	Triggers Transmit Descriptor Low Threshold Hit		0h	RW
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Triggers Receiver Timer Interrupt		0h	RW
06	RXO	Triggers Receiver Overrun. Set on receive data FIFO overrun		0h	RW
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Triggers Receive Descriptor Minimum Threshold hit		0h	RW
03	Rsvd	Reserved		0h	RW
02	Rsvd	Reserved. Must be written as '0'		0h	RV
01	TXQE	Triggers Transmit Queue Empty		0h	RW
00	TXDW	Triggers Transmit Descriptor Written Back		0h	RW

37.6.3.4 IMS0 – Interrupt 0 Mask Set/Read Register

This register contains which interrupts are enabled. An interrupt is enabled if its corresponding mask bit is set to 1, and disabled if its corresponding mask bit is set to 0. An interrupt is generated whenever one of the bits in this register is set, and the corresponding interrupt condition occurs, see the "ICR0 – Interrupt 0 Cause Read Register" on page 1454 for interrupt conditions.

A particular interrupt may be enabled by writing a 1 to the corresponding mask bit in this register. Any bits written with a 0, are unchanged. Thus, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear Register rather than writing a 0 to a bit in this register.

Table 37-40. IMS0: Interrupt 0 Mask Set/Read Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00D0h **Offset End:** 00D3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00D0h **Offset End:** 00D3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00D0h **Offset End:** 00D3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Enables Internal Bus Error		0h	RW
27	ERR_STAT	Enables Statistic Register ECC Error		0h	RW
26	ERR_MCFSPF	Enables Special Packet Filter Parity Error		0h	RW
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Enables DMA Packet Buffer ECC Error		0h	RW
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Enables DMA Transmit Descriptor Buffer ECC Error		0h	RW
20	ERR_RXDS	Enables DMA Receive Descriptor Buffer ECC Error		0h	RW
19 : 17	Rsvd	Reserved. Must be written as '0'		0h	RV
16	SRPD	Sets the mask for Small Receive Packet Detected and Transferred		0h	RW
15	TXD_LOW	Sets the mask for Transmit Descriptor Low Threshold Hit		0h	RW
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Sets the mask for Receiver Timer Interrupt		0h	RW
06	RXO	Sets the mask for Receiver Overrun. Set on receive data FIFO overrun		0h	RW
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Sets the mask for Receive Descriptor Minimum Threshold hit		0h	RW
03	Rsvd	Reserved		0h	RW

Table 37-40. IMS0: Interrupt 0 Mask Set/Read Register (Sheet 2 of 2)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 00D0h Offset End: 00D3h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 00D0h Offset End: 00D3h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 00D0h Offset End: 00D3h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	Rsvd	Reserved. Must be written as '0'		0h	RV
01	TXQE	Sets the mask for Transmit Queue Empty		0h	RW
00	TXDW	Sets the mask for Transmit Descriptor Written Back		0h	RW

37.6.3.5 IMC0 – Interrupt 0 Mask Clear Register

Software uses this register to disable an interrupt condition that was previously enabled. Interrupts are presented to the bus interface only when the mask bit is set and the interrupt condition is active. The status of the mask bit is reflected in the "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459, and the status of the cause bit is reflected in the "ICR0 – Interrupt 0 Cause Read Register" on page 1454. Software disables a given interrupt by writing a 1 to the corresponding bit in this register, a 0 is ignored. A read from this register will return the current value of the Interrupt mask settings, returning the same value as a read of the IMS register.

Table 37-41. IMC0: Interrupt 0 Mask Clear Register (Sheet 1 of 2)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 00D8h Offset End: 00DBh	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 00D8h Offset End: 00DBh	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 00D8h Offset End: 00DBh	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Clears the mask for Internal Bus Error		0h	WO
27	ERR_STAT	Clears the mask for Statistic Register ECC Error		0h	WO
26	ERR_MCFSPF	Clears the mask for the Filter Memory Errors		0h	WO
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Clears the mask for DMA Packet Buffer ECC Error		0h	WO

Table 37-41. IMC0: Interrupt 0 Mask Clear Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 00D8h **Offset End:** 00DBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 00D8h **Offset End:** 00DBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 00D8h **Offset End:** 00DBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Clears the mask for DMA Transmit Descriptor Buffer ECC Error		0h	WO
20	ERR_RXDS	Clears the mask for DMA Receive Descriptor Buffer ECC Error		0h	WO
19 : 17	Rsvd	Reserved		0h	RV
16	SRPD	Clears the mask for Small Receive Packet Detected and Transferred		0h	WO
15	TXD_LOW	Clears the mask for Transmit Descriptor Low Threshold Hit		0h	WO
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Clears the mask for Receiver Timer Interrupt		0h	WO
06	RXO	Clears the mask for Receiver Overrun. Set on receive data FIFO overrun		0h	WO
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Clears the mask for Receive Descriptor Minimum Threshold hit		0h	WO
03	Rsvd	Reserved		0h	WO
02	Rsvd	Reserved. Must be written as ‘0’		0h	RV
01	TXQE	Clears the mask for Transmit Queue Empty		0h	WO
00	TXDW	Clears the mask for Transmit Descriptor Written Back		0h	WO

37.6.3.6 ICR1 – Interrupt 1 Cause Read Register

This register contains all interrupt conditions for the GbE. Whenever an interrupt causing event occurs, the corresponding interrupt bit is set in this register. See "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 for additional details.

Note: All register bits clear on read. Thus, reading this register implicitly acknowledges all pending interrupt events. Writing a 1 to any bit in the register will also clear that bit. Writing a 0 to any bit will have no effect.

Table 37-42. ICR1: Interrupt 1Cause Read Register (Sheet 1 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	RSVD	Reserved		0h	RV
28	ERR_INTBUS	**Internal Bus Error.** This bit indicates that an error occurred during either a Target or Host transaction on the bus. Refer to Section 37.5.12, "Error Handling" for complete details. The details of this error are reported in the INTBUS_ERR_STAT register.		0h	RCWC
27	ERR_STAT	**Statistic Register ECC Error.** The Statistic Registers are implemented using a memory that uses a single-bit correct/multi-bit detect ECC parity algorithm to protect it. This bit indicates that a multi-bit error has occurred on a read from that memory. No indication of a single-bit error correction will be given by hardware. ***Note:*** *If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)*		0h	RCWC
26	ERR_MCFSPF	**Multicast Filter Parity Error/Special Packet Filter Parity Error.** The Multicast Filter and Special Packets Filter use parity protected SRAMs for data buffers. This bit indicates that a parity error has occurred on a read from either of these data buffers. This error is considered non-fatal and will clear after a read of the MEM_ERR_STAT register.		0h	RCWC
25 : 24	RSVD	Reserved		0h	RV
23	ERR_PB	**DMA Packet Buffer 2-bit ECC Error.** The 64KB DMA Packet Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for data. This bit indicates that a multi-bit error has occurred on a read from that SRAM. No indication of a single-bit error correction will be given by hardware. ***Note:*** *If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).*		0h	RCWC
22	RSVD	Reserved		0h	RV

Table 37-42. ICR1: Interrupt 1Cause Read Register (Sheet 2 of 3)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08C0h **Offset End:** 08C3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
21	ERR_TXDS	DMA Transmit Descriptor 2-bit ECC Error. The DMA Transmit Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).		0h	RCWC
20	ERR_RXDS	DMA Receive Descriptor 2-bit ECC Error. The DMA Receive Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).		0h	RCWC
19 : 17	RSVD	Reserved		0h	RV
16	SRPD	**Small Receive Packet Detected.** Indicates that a packet of size RSRPD.SIZE register has been detected and transferred to host memory. The interrupt is only asserted if RSRPD.SIZE register has a non-zero value		0h	RCWC
15	TXD_LOW	Transmit Descriptor Low Threshold hit. Indicates that the descriptor ring has reached the threshold specified in "TXDCTL – Transmit Descriptor Control Register" on page 1500.		0h	RCWC
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	**Receiver Timer Interrupt.** Set when the timers expire, see "Receive Interrupts" on page 1360 for details.		0h	RCWC
06	RXO	**Receiver Overrun.** Set on receive data FIFO overrun. Could be caused either because there are no available buffers or because Internal Bus receive bandwidth is inadequate.		0h	RCWC
05	Rsvd	Reserved		0h	RV
04	RXDMT0	**Receive Descriptor Minimum Threshold Hit.** Indicates that the minimum number of receive descriptors are available and software should load more receive descriptors.		0h	RCWC
03	Rsvd	Reserved		0h	RCWC
02	RSVD	Reserved		0h	RV

Table 37-42. ICR1: Interrupt 1Cause Read Register (Sheet 3 of 3)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 08C0h Offset End: 08C3h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 08C0h Offset End: 08C3h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 08C0h Offset End: 08C3h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
01	TXQE	**Transmit Queue Empty.** Set when the last descriptor block for a transmit queue has been used.		0h	RCWC
00	TXDW	**Transmit Descriptor Written Back.** Set when hardware processes a descriptor with its RS bit set. If using delayed interrupts (TDESC.IDE is set in the Transmit Descriptor CMD), the interrupt is delayed until after one of the delayed-timers (TIDV or TADV) expires.		0h	RCWC

37.6.3.7 ICS1 – Interrupt 1 Cause Set Register

Software uses this register to set an interrupt condition. Assuming the interrupt mask is set, any bit written with a 1 triggers the corresponding interrupt, see "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 and "ICR0 – Interrupt 0 Cause Read Register" on page 1454.

Table 37-43. ICS1: Interrupt 0 Cause Set Register (Sheet 1 of 2)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 08C8h Offset End: 08CBh	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 08C8h Offset End: 08CBh	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 08C8h Offset End: 08CBh	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Triggers Internal Bus Error		0h	RW
27	ERR_STAT	Triggers Statistic Register ECC Error		0h	RW
26	ERR_MCFSPF	Triggers Special Packet Filter Parity Error		0h	RW
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Triggers DMA Packet Buffer ECC Error		0h	RW

Table 37-43. ICS1: Interrupt 0 Cause Set Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08C8h **Offset End:** 08CBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08C8h **Offset End:** 08CBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08C8h **Offset End:** 08CBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Triggers DMA Transmit Descriptor Buffer ECC Error		0h	RW
20	ERR_RXDS	Triggers DMA Receive Descriptor Buffer ECC Error		0h	RW
19 : 17	Rsvd	Reserved		0h	RV
16	SRPD	Triggers Small Receive Packet Detected and Transferred		0h	RW
15	TXD_LOW	Triggers Transmit Descriptor Low Threshold Hit		0h	RW
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Triggers Receiver Timer Interrupt		0h	RW
06	RXO	Triggers Receiver Overrun. Set on receive data FIFO overrun		0h	RW
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Triggers Receive Descriptor Minimum Threshold hit		0h	RW
03	Rsvd	Reserved		0h	RV
02	Rsvd	Reserved. Must be written as ‘0’		0h	RV
01	TXQE	Triggers Transmit Queue Empty		0h	RW
00	TXDW	Triggers Transmit Descriptor Written Back		0h	RW

37.6.3.8 IMS1 – Interrupt 1 Mask Set/Read Register

This register contains which interrupts are enabled. An interrupt is enabled if its corresponding mask bit is set to 1, and disabled if its corresponding mask bit is set to 0. An interrupt is generated whenever one of the bits in this register is set, and the corresponding interrupt condition occurs, see the "ICR0 – Interrupt 0 Cause Read Register" on page 1454 for interrupt conditions.

A particular interrupt may be enabled by writing a 1 to the corresponding mask bit in this register. Any bits written with a 0, are unchanged. Thus, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear Register rather than writing a 0 to a bit in this register.

Table 37-44. IMS1: Interrupt 1 Mask Set/Read Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Enables Internal Bus Error	RW	0h	WO
27	ERR_STAT	Enables Statistic Register ECC Error	RW	0h	WO
26	ERR_MCFSPF	Enables Special Packet Filter Parity Error	RW	0h	WO
25 : 24	Rsvd	Reserved	RV	0h	RV
23	ERR_PKBUF	Enables DMA Packet Buffer ECC Error	RW	0h	WO
22	Rsvd	Reserved	RV	0h	RV
21	ERR_TXDS	Enables DMA Transmit Descriptor Buffer ECC Error	RW	0h	WO
20	ERR_RXDS	Enables DMA Receive Descriptor Buffer ECC Error	RW	0h	WO
19 : 17	Rsvd	Reserved		0h	RV
16	SRPD	Sets the mask for Small Receive Packet Detected and Transferred		0h	RW
15	TXD_LOW	Sets the mask for Transmit Descriptor Low Threshold Hit		0h	RW
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Sets the mask for Receiver Timer Interrupt		0h	RW
06	RXO	Sets the mask for Receiver Overrun. Set on receive data FIFO overrun		0h	RW
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Sets the mask for Receive Descriptor Minimum Threshold hit		0h	RW
03	Rsvd	Reserved		0h	RW

Table 37-44. IMS1: Interrupt 1 Mask Set/Read Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08D0h **Offset End:** 08D3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	Rsvd	Reserved. Must be written as '0'		0h	RW
01	TXQE	Sets the mask for Transmit Queue Empty		0h	RW
00	TXDW	Sets the mask for Transmit Descriptor Written Back		0h	RW

37.6.3.9 IMC1 – Interrupt 1 Mask Clear Register

Software uses this register to disable an interrupt condition that was previously enabled. Interrupts are presented to the bus interface only when the mask bit is set and the interrupt condition is active. The status of the mask bit is reflected in the "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459, and the status of the cause bit is reflected in the "ICR0 – Interrupt 0 Cause Read Register" on page 1454. Software disables a given interrupt by writing a 1 to the corresponding bit in this register, a 0 is ignored.

Table 37-45. IMC1: Interrupt 1 Mask Clear Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08D8h **Offset End:** 08DBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08D8h **Offset End:** 08DBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08D8h **Offset End:** 08DBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Clears the mask for Internal Bus Error		0h	WO
27	ERR_STAT	Clears the mask for Statistic Register ECC Error		0h	WO
26	ERR_MCFSPF	Clears the mask for the Filter Memory Errors		0h	WO
25 : 24	Rsvd	Reserved		0h	RV

Table 37-45. IMC1: Interrupt 1 Mask Clear Register (Sheet 2 of 2)

Description:			
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:0:0	**Offset Start:** 08D8h **Offset End:** 08DBh
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:1:0	**Offset Start:** 08D8h **Offset End:** 08DBh
View: PCI 3	**BAR:** CSRBAR	**Bus:Device:Function:** M:2:0	**Offset Start:** 08D8h **Offset End:** 08DBh
Size: 32 bits	**Default:** 00000000h		**Power Well:** GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
23	ERR_PKBUF	Clears the mask for DMA Packet Buffer ECC Error		0h	WO
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Clears the mask for DMA Transmit Descriptor Buffer ECC Error		0h	WO
20	ERR_RXDS	Clears the mask for DMA Receive Descriptor Buffer ECC Error		0h	WO
19 : 17	Rsvd	Reserved		0h	RV
16	SRPD	Clears the mask for Small Receive Packet Detected and Transferred		0h	WO
15	TXD_LOW	Clears the mask for Transmit Descriptor Low Threshold Hit		0h	WO
14 : 8	Rsvd	Reserved		0h	RV
07	RXT0	Clears the mask for Receiver Timer Interrupt		0h	WO
06	RXO	Clears the mask for Receiver Overrun. Set on receive data FIFO overrun		0h	WO
05	Rsvd	Reserved		0h	RV
04	RXDMT0	Clears the mask for Receive Descriptor Minimum Threshold hit		0h	WO
03	Rsvd	Reserved		0h	WO
02	Rsvd	Reserved		0h	RV
01	TXQE	Clears the mask for Transmit Queue Empty		0h	WO
00	TXDW	Clears the mask for Transmit Descriptor Written Back		0h	WO

37.6.3.10 ICR2 – Error Interrupt Cause Read Register

This register contains all interrupt conditions for the GbE. Whenever an interrupt causing event occurs, the corresponding interrupt bit is set in this register. See "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 for additional details.

Note: All register bits clear on read. Thus, reading this register implicitly acknowledges all pending interrupt events. Writing a 1 to any bit in the register will also clear that bit. Writing a 0 to any bit will have no effect.

Table 37-46. ICR2: Error Interrupt Cause Read Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 29	RSVD	Reserved		0h	RV
28	ERR_INTBUS	**Internal Bus Error.** This bit indicates that an error occurred during either a Target or Host transaction on the bus. Refer to Section 37.5.12, "Error Handling" for complete details. The details of this error are reported in the INTBUS_ERR_STAT register.		0h	RCWC
27	ERR_STAT	**Statistic Register ECC Error.** The Statistic Registers are implemented using a memory that uses a single-bit correct/multi-bit detect ECC parity algorithm to protect it. This bit indicates that a multi-bit error has occurred on a read from that memory. No indication of a single-bit error correction will be given by hardware. ***Note:*** *If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST)*		0h	RCWC
26	ERR_MCFSPF	**Multicast Filter Parity Error/Special Packet Filter Parity Error.** The Multicast Filter and Special Packets Filter use parity protected SRAMs for data buffers. This bit indicates that a parity error has occurred on a read from either of these data buffers.		0h	RCWC
25 : 24	RSVD	Reserved		0h	RV
23	ERR_PB	**DMA Packet Buffer 2-bit ECC Error.** The 64KB DMA Packet Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for data. This bit indicates that a multi-bit error has occurred on a read from that SRAM. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).		0h	RCWC
22	RSVD	Reserved		0h	RV

Table 37-46. ICR2: Error Interrupt Cause Read Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08E0h **Offset End:** 08E3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
21	ERR_TXDS	DMA Transmit Descriptor 2-bit ECC Error. The DMA Transmit Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).		0h	RCWC
20	ERR_RXDS	DMA Receive Descriptor 2-bit ECC Error. The DMA Receive Descriptor Buffer uses a single-bit correct/multi-bit detect ECC parity algorithm to protect the SRAM it uses for a data buffer. This bit indicates that a multi-bit error has occurred on a read from that data buffer. No indication of a single-bit error correction will be given by hardware. ***Note:*** If this interrupt asserts, further GbE DMA Reads and Writes are blocked until software issues a soft reset to the GbE by writing the Device Control Register (CTRL.RST).		0h	RCWC
19 : 00	RSVD	Reserved		0h	RV

37.6.3.11 ICS2 – Error Interrupt Cause Set Register

Software uses this register to set an interrupt condition. Assuming the interrupt mask is set, any bit written with a 1 triggers the corresponding interrupt, see "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459 and "ICR0 – Interrupt 0 Cause Read Register" on page 1454.

Table 37-47. ICS2: Error Interrupt Cause Set Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08E8h **Offset End:** 08EBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08E8h **Offset End:** 08EBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08E8h **Offset End:** 08EBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Triggers Internal Bus Error		0h	RW
27	ERR_STAT	Triggers Statistic Register ECC Error		0h	RW
26	ERR_MCFSPF	Triggers Special Packet Filter Parity Error		0h	RW
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Triggers DMA Packet Buffer ECC Error		0h	RW
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Triggers DMA Transmit Descriptor Buffer ECC Error		0h	RW
20	ERR_RXDS	Triggers DMA Receive Descriptor Buffer ECC Error		0h	RW
19 : 00	Rsvd	Reserved		0h	RV

37.6.3.12 IMS2 – Error Interrupt Mask Set/Read Register

This register contains which interrupts are enabled. An interrupt is enabled if its corresponding mask bit is set to 1, and disabled if its corresponding mask bit is set to 0. An interrupt is generated whenever one of the bits in this register is set, and the corresponding interrupt condition occurs, see the "ICR0 – Interrupt 0 Cause Read Register" on page 1454 for interrupt conditions.

A particular interrupt may be enabled by writing a 1 to the corresponding mask bit in this register. Any bits written with a 0, are unchanged. Thus, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear Register rather than writing a 0 to a bit in this register.

Table 37-48. IMS2: Error Interrupt Mask Set/Read Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08F0h **Offset End:** 08F3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08F0h **Offset End:** 08F3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08F0h **Offset End:** 08F3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Enables Internal Bus Error		0h	RW
27	ERR_STAT	Enables Statistic Register ECC Error		0h	RW
26	ERR_MCFSPF	Enables Special Packet Filter Parity Error		0h	RW
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Enables DMA Packet Buffer ECC Error		0h	RW
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Enables DMA Transmit Descriptor Buffer ECC Error		0h	RW
20	ERR_RXDS	Enables DMA Receive Descriptor Buffer ECC Error		0h	RW
19 : 00	Rsvd	Reserved		0h	RV

37.6.3.13 IMC2 – Error Interrupt Mask Clear Register

Software uses this register to disable an interrupt condition that was previously enabled. Interrupts are presented to the bus interface only when the mask bit is set and the interrupt condition is active. The status of the mask bit is reflected in the "IMS0 – Interrupt 0 Mask Set/Read Register" on page 1459, and the status of the cause bit is reflected in the "ICR0 – Interrupt 0 Cause Read Register" on page 1454. Software disables a given interrupt by writing a 1 to the corresponding bit in this register, a 0 is ignored.

Table 37-49. IMC2: Error Interrupt Mask Clear Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 08F8h **Offset End:** 08FBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 08F8h **Offset End:** 08FBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 08F8h **Offset End:** 08FBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 29	Rsvd	Reserved		0h	RV
28	ERR_INTBUS	Clears the mask for Internal Bus Error		0h	WO
27	ERR_STAT	Clears the mask for Statistic Register ECC Error		0h	WO
26	ERR_INT	Clears the mask for Internal Memory Error		0h	WO
25 : 24	Rsvd	Reserved		0h	RV
23	ERR_PKBUF	Clears the mask for DMA Packet Buffer ECC Error		0h	WO
22	Rsvd	Reserved		0h	RV
21	ERR_TXDS	Clears the mask for DMA Transmit Descriptor Buffer ECC Error		0h	WO
20	ERR_RXDS	Clears the mask for DMA Receive Descriptor Buffer ECC Error		0h	WO
19 : 00	Rsvd	Reserved		0h	RV

37.6.4 Receive Registers: Detailed Descriptions

37.6.4.1 RCTL – Receive Control Register

This register controls the types and sizes of packets received, as well as any manipulation of those received packets. The size of the receive buffers where those packets reside before they are transferred to system memory is also controlled here.

Table 37-50. RCTL: Receive Control Register (Sheet 1 of 4)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0100h **Offset End:** 0103h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 27	Rsvd	Reserved		0h	RV
26	SECRC	**Strip Ethernet CRC.** This bit controls whether the hardware strips the Ethernet CRC from the received packet. This stripping occurs prior to any checksum calculations. The stripped CRC is not DMA'd to host memory and is not included in the length reported in the descriptor.		0h	RW
25	BSEX	**Buffer Size Extension.** Combined with RCTL.BSIZE to program the receive buffer size. Control of receive buffer size permits software to trade-off descriptor performance versus required storage space. Buffers that are 2048 bytes require only one descriptor per receive packet maximizing descriptor efficiency. Buffers that are 256 bytes maximize memory efficiency at a cost of multiple descriptors for packets longer than 256 bytes. RCTL.BSEX = 0 / RCTL.BSIZE = 00 -> Receive Buffer Size = 2048B RCTL.BSEX = 0 / RCTL.BSIZE = 01 -> Receive Buffer Size = 1024B RCTL.BSEX = 0 / RCTL.BSIZE = 10 -> Receive Buffer Size = 512B RCTL.BSEX = 0 / RCTL.BSIZE = 11 -> Receive Buffer Size = 256B RCTL.BSEX = 1 / RCTL.BSIZE = 00 -> Reserved RCTL.BSEX = 1 / RCTL.BSIZE = 01 -> Receive Buffer Size = 16384B RCTL.BSEX = 1 / RCTL.BSIZE = 10 -> Receive Buffer Size = 8192B RCTL.BSEX = 1 / RCTL.BSIZE = 11 -> Receive Buffer Size = 4096B		0h	RW
24	Rsvd	Reserved		0h	RV
23	PMCF	Pass MAC Control Frames. This bit controls the DMA function of MAC control frames (other than flow control). A MAC control frame in this context must be addressed to either the MAC control frame multicast address or the station address, it must match the type field and must NOT match the PAUSE opcode of 0x0001. 0 = Do not pass MAC control frames 1 = Pass any MAC control frame (type field value of 0x8808) that does not contain the pause opcode of 0x0001.		0h	RW

Table 37-50. RCTL: Receive Control Register (Sheet 2 of 4)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0100h **Offset End:** 0103h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
22	DPF	**Discard Pause Frames.** This bit controls the DMA function of flow control packets addressed to the station address (RAH/RAL[0]). If a packet is a valid flow control packet and is addressed to the station address it will not be DMA'd to host memory if RCTL.DPF=1. 0 = Incoming frames are subject to filter comparison 1 = Incoming valid PAUSE frames discarded even if they match any of the filter registers		0h	RW
21	Rsvd	Reserved		0h	RV
20	CFI	Canonical Form Indicator. One of the three bits that control the VLAN filter table. This bit may be compared to the CFI bit found in the 802.1q packet as part of the acceptance criteria. RCTL.CFIEN and RCTL.VFE determine whether or not this comparison takes place.		0h	RW
19	CFIEN	**Canonical Form Indicator Enable.** One of the three bits that control the VLAN filter table. This bit enables using the CFI bit found in the 802.1q packet as part of the acceptance criteria. The next two are used to decide whether the CFI bit found in the.1Q packet should be used as part of the acceptance criteria. 0 = CFI Disabled: bit not compared to determine packet acceptance 1 = CFI from packet must match CFI field for acceptance of 802.1q packet		0h	RW
18	VFE	VLAN Filter Enable. One of the three bits that control the VLAN filter table. This bit determines whether the table participates in the packet acceptance criteria. 0 = Disabled, filter table does not decide packet acceptance 1 = Enabled, filter table decides acceptance of 802.1q packets		0h	RW

Table 37-50. RCTL: Receive Control Register (Sheet 3 of 4)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0100h **Offset End:** 0103h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
17 : 16	BSIZE	**Receive Buffer Size.** Combined with RCTL.BSEX to program the receive buffer size. Control of receive buffer size permits software to trade-off descriptor performance versus required storage space. Buffers that are 2048 bytes require only one descriptor per receive packet maximizing descriptor efficiency. Buffers that are 256 bytes maximize memory efficiency at a cost of multiple descriptors for packets longer than 256 bytes. RCTL.BSEX = 0 / RCTL.BSIZE = 00 -> Receive Buffer Size = 2048B RCTL.BSEX = 0 / RCTL.BSIZE = 01 -> Receive Buffer Size = 1024B RCTL.BSEX = 0 / RCTL.BSIZE = 10 -> Receive Buffer Size = 512B RCTL.BSEX = 0 / RCTL.BSIZE = 11 -> Receive Buffer Size = 256B RCTL.BSEX = 1 / RCTL.BSIZE = 00 -> Reserved RCTL.BSEX = 1 / RCTL.BSIZE = 01 -> Receive Buffer Size = 16384B RCTL.BSEX = 1 / RCTL.BSIZE = 10 -> Receive Buffer Size = 8192B RCTL.BSEX = 1 / RCTL.BSIZE = 11 -> Receive Buffer Size = 4096B		00h	RW
15	BAM	Broadcast Accept Mode. 0 = Ignore broadcast (unless it matches exact or imperfect filters) 1 = Accept broadcast packets		0h	RW
14	Rsvd	Reserved		0h	RV
13 : 12	MO	**Multicast Offset.** This determines which bits of the incoming multicast address are used in looking up the bit vector. • 00 = [47:36] • 01 = [46:35] • 10 = [45:34] • 11 = [43:32]		0h	RW
11 : 10	Rsvd	Reserved		0h	RV
09 : 08	RDMTS	**Receive Descriptor Minimum Threshold Size.** These bits determines the threshold value for free receive descriptors. The corresponding interrupt is set whenever the fractional number of free descriptors becomes equal to RCTL.RDMTS. Refer to "RDLEN – Receive Descriptor Length Register" on page 1481 for further information. • 00 = 1/2 • 01 = 1/4 • 10 = 1/8 • 11 = Reserved		00h	RW

Table 37-50. RCTL: Receive Control Register (Sheet 4 of 4)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0100h **Offset End:** 0103h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0100h **Offset End:** 0103h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
07 : 06	LBM	Loopback mode. These bits enable the loopback function.When using a PHY, a value of 00 should be used and the PHY is configured for loopback through the MDIO interface. • 00 = Normal operation (or PHY loopback in GMII/MII mode) • 01 = MAC Loopback enable (only supported for GMII/MII mode) • 10 = Reserved • 11 = Reserved • 11 = Reserved ***Note:*** PHY devices require programming for loopback operation using MDIO accesses. ***Note:*** The GbE must be configured for Full-Duplex operation if Mac Loopback mode is enabled.		00h	RW
05	LPE	Long packet enable. This bit controls whether long packet reception is permitted. 0 = Disabled, hardware discards packets longer than 1522B 1 = Enabled, 16384B is the maximum packet size that the GbE can receive		0h	RW
04	MPE	Multicast promiscuous enable. 0 = Disabled 1 = Enabled		0h	RW
03	UPE	Unicast promiscuous enable. 0 = Disabled 1 = Enabled		0h	RW
02	SBP	Store bad packets. 0 = Disabled 1 = Enabled		0h	RW
01	EN	Receiver Enable. 0 = All incoming packets are immediately dropped and are not stored in the receive FIFO. If a packet is already in-progress when disabled it will be finished. 1 = Incoming packet reception is enabled.		0h	RW
00	Rsvd	Reserved		0h	RV

37.6.4.2 FCRTL: Flow Control Receive Threshold Low Register

This register contains the receive threshold used to determine when to send an XON packet, counting in units of bytes. The lower 3 bits must be programmed to 0 (8B granularity). Whenever hardware crosses the receive high threshold (becoming more full), and then crosses the receive low threshold, then hardware will transmits an XON frame (if enabled with FCRTL.XONE).

Note: Flow control reception/transmission are negotiated capabilities by the Auto-Negotiation process. When the device is manually configured, flow control operation is determined by the CTRL.RFCE & CTRL.TFCE.

Table 37-51. FCRTL: Flow Control Receive Threshold Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2160h **Offset End:** 2163h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2160h **Offset End:** 2163h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2160h **Offset End:** 2163h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	XONE	XON Enable 0b = Disabled. 1b = Enabled. When set, enables the Ethernet controller to transmit XON packets based on receive FIFO crosses FCRTL.RTL threshold value, or based on external pins XOFF and XON. See Section 37.6.4.3, "FCRTH – Flow Control Receive Threshold High Register" on page 1479		0h	RW
30 : 16	Rsvd	Reserved		0h	RV
15 : 03	RTL	**Receive Threshold Low.** FIFO low water mark for flow control transmission.		0h	RW
02 : 00	0	Writes are ignored, reads return 0.		0h	RV

37.6.4.3 FCRTH – Flow Control Receive Threshold High Register

This register contains the receive threshold used to determine when to send an XOFF packet. It counts in units of bytes. This value must be at least 8 bytes less than the maximum number of bytes allocated to the Receive Packet Buffer (PBA, RXA), and the lower 3 bits must be programmed to 0 (8B granularity). Whenever the receive FIFO reaches the fullness indicated by FCRTH.RTH, hardware transmits a PAUSE frame if the transmission of flow control frames is enabled.

Note: Flow control reception/transmission are negotiated capabilities by the Auto-Negotiation process. When the device is manually configured, flow control operation is determined by the CTRL.RFCE & CTRL.TFCE.

Table 37-52. FCRTH: Flow Control Receive Threshold High Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2168h **Offset End:** 216Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2168h **Offset End:** 216Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2168h **Offset End:** 216Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31	XFCE	External Flow Control Enabled 0b = Disabled. 1b = Enabled. Allows the Ethernet controller to send XOFF and XON frames based on external pins XOFF and XON. The transmission of pause frames must be also enabled through the CTRL.TFCE control bit. When the XOFF signal is asserted high, the Ethernet controller transmits a single XOFF frame. The assertion of XON (after deassertion of XOFF) initiates an XON frame transmission, if enabled by FCRTL.XONE. The assertion/deassertion of XON is required between assertions of XOFF in order to send another XOFF frame. This behavior also provides a built-in hysteresis mechanism. ***Note:*** The EP80579 does not have external XON/XOFF pins and therefore does not support external flow control enable. This bit must be set to 0 for correct operation.		0h	RW
30 : 16	Rsvd	Reserved		0h	RV
15 : 03	RTH	**Receive Threshold High.** FIFO high water mark for flow control transmission.		0h	RW
02 : 00	0	Writes are ignored, reads return 0.		0h	RV

37.6.4.4 RDBAL – Receive Descriptor Base Address Low Register

This register contains the lower bits of the 64 bit descriptor base address. The Receive Descriptor Base Address must point to a 16B aligned block of data (i.e. the lower 4 bits are always 0).

Table 37-53. RDBAL: Receive Descriptor Base Address Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2800h **Offset End:** 2803h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2800h **Offset End:** 2803h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2800h **Offset End:** 2803h	
Size: 32 bits	**Default:** XXXXXXX0h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	RDBAL	Receive Descriptor Base Address Low		X	RW
03 : 00	0	Writes are ignored, reads return 0.		0h	RV

37.6.4.5 RDBAH – Receive Descriptor Base Address High Register

This register contains the upper 32 bits of the 64 bit Descriptor base address.

Table 37-54. RDBAH: Receive Descriptor Base Address High Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2804h **Offset End:** 2807h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2804h **Offset End:** 2807h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2804h **Offset End:** 2807h	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RDBAH	Receive Descriptor Base Address. ***Note:*** RDBAH[31:0] must be set to 0.		X	RW

37.6.4.6 RDLEN – Receive Descriptor Length Register

This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. This value must be 128B aligned (i.e., the lower 7 bits are always 0).

Table 37-55. RDLEN: Receive Descriptor Length Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2808h **Offset End:** 280Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2808h **Offset End:** 280Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2808h **Offset End:** 280Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Rsvd	Reserved		0h	RV
19 : 07	LEN	Descriptor Length		0h	RW
06 : 00	0	Writes are ignored, reads return 0.		0h	RV

37.6.4.7 RDH – Receive Descriptor Head Register

This register contains the head pointer for the receive descriptor buffer. The register points to a 16B datum. Hardware controls the pointer. Writing this register at any time will cause indeterminate behavior.

Table 37-56. RDH: Receive Descriptor Head Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2810h **Offset End:** 2813h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2810h **Offset End:** 2813h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2810h **Offset End:** 2813h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	RDH	Receive Descriptor Head		0h	RW

37.6.4.8 RDT – Receive Descriptor Tail Register

This register contains the tail pointers for the receive descriptor buffer. The register points to a 16B datum. Software writes the tail register to add receive descriptors to the hardware free list for the ring.

Table 37-57. RDT: Receive Descriptor Tail Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2818h **Offset End:** 281Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2818h **Offset End:** 281Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2818h **Offset End:** 281Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Core Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	RDT	Receive Descriptor Tail		0h	RW

37.6.4.9 RDTR – RX Interrupt Delay Timer (Packet Timer) Register

This register is used to delay interrupt notification for the receive descriptor ring by coalescing interrupts for multiple received packets. Delaying interrupt notification helps maximize the number of receive packets serviced by a single interrupt.

This feature operates by initiating a countdown timer upon successfully receiving each packet to system memory. If a subsequent packet is received BEFORE the timer expires, the timer is reinitialized to the programmed value and re-starts its countdown. If the timer expires due to NOT having received a subsequent packet within the programmed interval, pending receive descriptor writebacks are flushed and a receive timer interrupt is generated.

Setting the value to 0b represents no delay from a receive packet to the interrupt notification, an results in immediate interrupt notification for each received packet.

Writing this register with FPD set initiates an immediate expiration of the timer, causing a writeback of any consumed receive descriptors pending writeback, and results in a receive timer interrupt in the ICR.

Receive interrupts due to a Receive Absolute Timer (RADV) expiration cancels a pending RDTR interrupt. The RDTR countdown timer is reloaded but halted, so as to avoid generation of a spurious second interrupt after the RADV has been noted, but might be restarted by a subsequent received packet.

Table 37-58. RDTR: RX Interrupt Delay Timer (Packet Timer) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2820h **Offset End:** 2823h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2820h **Offset End:** 2823h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2820h **Offset End:** 2823h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	FPD	**Flush Partial Descriptor.** Writing this bit with 1 initiates an immediate expiration of the timer, causing a writeback of any consumed receive descriptors pending writeback, and results in a receive timer interrupt in the ICR register. This bit is self clearing and always reads 0.		0h	WO
30 : 16	Rsvd	Reserved		0h	RV
15 : 00	RPDT	**Receive Packet Delay Timer** Timer increments are RMII: 1.28 microseconds RGMII: 1.024 microseconds. See register description above		0h	RW

37.6.4.10 RXDCTL – Receive Descriptor Control Register

This register controls the fetching and write-back of receive descriptors. The three threshold values are used to determine when descriptors will be read from and written to host memory. The values may be in units of cache lines or 16B descriptors.

Table 37-59. RXDCTL: Receive Descriptor Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2828h **Offset End:** 282Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2828h **Offset End:** 282Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2828h **Offset End:** 282Bh	
Size: 32 bits	**Default:** 00010000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 25	Rsvd	Reserved		0h	RV

Table 37-59. RXDCTL: Receive Descriptor Control Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2828h **Offset End:** 282Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2828h **Offset End:** 282Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2828h **Offset End:** 282Bh	
Size: 32 bits	**Default:** 00010000h			**Power Well:**	GbE0: Core Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
24	GRAN	**Granularity** of the thresholds in this register. 0 = Threshold values are in units of Cache Lines, thresholds specified must not be greater than 31 descriptors (496B) or 15 32B cache lines. 1 = Threshold values are in units of Descriptors (16B each)		0h	RW
23 : 22	Rsvd	Reserved		0h	RV
21 : 16	WTHRESH	**Write-back Threshold.** This field controls the write-back of processed receive descriptors. This threshold refers to the number of receive descriptors in the GbE hardware buffer which are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back will occur only after more than WTHRESH descriptors are available for write-back. ***Note:*** Since the default value for this field is 1, the descriptors are normally written back as soon as one cache line is available. This field must contain a non-zero value to take advantage of the write-back bursting capabilities of the EP80579's GbE.		01h	RW
15 : 14	Rsvd	Reserved		0h	RV
13 : 08	HTHRESH	**Host Threshold.** This field is used to control the fetching of descriptors from host memory. This threshold refers to the number of valid, unprocessed receive descriptors that must exist in host memory before they will be fetched.		0h	RW
07 : 06	Rsvd	Reserved		0h	RV
05 : 00	PTHRESH	**Prefetch Threshold.** This field is used to control when a prefetch of descriptors will be considered. This threshold refers to the number of valid, unprocessed receive descriptors the chip has in its GbE hardware buffer. If this number drops below PTHRESH, the algorithm will consider pre-fetching descriptors from host memory. This fetch will not happen however unless there are at least HTHRESH valid descriptors in host memory to fetch.		0h	RW

37.6.4.11 RADV – Receive Interrupt Absolute Delay Timer Register

Table 37-60. RADV: Receive Interrupt Absolute Delay Timer Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 282Ch **Offset End:** 282Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 282Ch **Offset End:** 282Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 282Ch **Offset End:** 282Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	RADT	**Receive Absolute Delay Timer** Receive Absolute delay timer measured in increments of RMII: 1.28 microseconds RGMII: 1.024 microseconds. (0b = disabled) If the packet delay timer is used to coalesce receive interrupts, the Ethernet controller ensures that when receive traffic abates, an interrupt is generated within a specified interval of no receives. During times when receive traffic is continuous, it may be necessary to ensure that no receive remains unnoticed for too long an interval. This register can be used to ENSURE that a receive interrupt occurs at some predefined interval after the first packet is received. When this timer is enabled, a separate absolute countdown timer is initiated upon successfully receiving each packet to system memory. When this absolute timer expires, pending receive descriptor writebacks are flushed and a receive timer interrupt is generated. Setting this register to 0b disables the absolute timer mechanism (the RDTR register should be used with a value of 0b to cause immediate interrupts for all receive packets). Receive interrupts due to a Receive Packet Timer (RDTR) expiration cancels a pending RADV interrupt. If enabled, the RADV countdown timer is reloaded but halted, so as to avoid generation of a spurious second interrupt after the RDTR has been noted.		0h	RW

37.6.4.12 RSRPD – Receive Small Packet Detect Interrupt Register

This register will generate an interrupt condition when any received packet is less than or equal to the size programmed.

Table 37-61. RSRPD: Receive Small Packet Detect Interrupt Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 2C00h **Offset End:** 2C03h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 2C00h **Offset End:** 2C03h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 2C00h **Offset End:** 2C03h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	Rsvd	Reserved		0h	RV
11 : 00	SIZE	Any packet received that is <= SIZE will assert an interrupt condition (ICR.SRPD). This field is specified in bytes and includes the headers and the CRC but not the VLAN header in the size calculation.		0h	RW

37.6.4.13 RXCSUM – Receive Checksum Control Register

This register controls the receive checksum off loading features. The GbE supports the off loading of three receive checksum calculations: the Packet Checksum, the IP Header Checksum, and the TCP/UDP Checksum. Supported frame types are Ethernet II and Ethernet SNAP. This register should only be written when the receiver is not enabled (i.e., only when RCTL.EN = 0).

Table 37-62. RXCSUM: Receive Checksum Control Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5000h **Offset End:** 5003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5000h **Offset End:** 5003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5000h **Offset End:** 5003h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 10	Rsvd	Reserved		0h	RV
09	TUOFL	**TCP/UDP Checksum Off load Enable.** This bit is used to enable the TCP/UDP Checksum off-loading feature. 0 = TCP/UDP Checksum Off load Disabled 1 = Hardware will calculate the TCP or UDP checksum and indicate a pass/fail indication to software via the TCP/UDP Checksum Error bit (TCPE).		0h	RW
08	IPOFL	**IP Checksum Off load Enable.** This bit is used to enable the IP Checksum off-loading feature. 0 = IP Checksum Off load Disabled 1 = Hardware will calculate the IP checksum and indicate a pass/fail indication to software via the IP Checksum Error bit (IPE) in the ERROR field of the receive descriptor.		0h	RW
07 : 00	PCSS	**Packet Checksum Start.** This field controls the starting byte for the Packet Checksum calculation. The Packet Checksum is the one's complement over the receive packet, starting from the byte indicated by PCSS (0 corresponds to the first byte of the packet), after stripping. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and with PCSS set to 14, the packet checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, Type and Length) and the 4-byte VLAN tag. The Packet Checksum will not include the Ethernet CRC if the RCTL.SECRC bit is set. Software must make the required offsetting computation (to back out the bytes that should not have been included and to include the pseudo-header) prior to comparing the Packet Checksum against the TCP checksum stored in the packet.		0h	RW

37.6.4.14 MTA[0-127] – 128 Multicast Table Array Registers

There is one register per 32 bits of the Multicast Address Table for a total of 128 registers (thus the MTA[127:0] designation). The size of the word array depends on the number of bits implemented in the multicast address table. Software must mask to the desired bit on reads and supply a 32-bit word on writes. Refer to "Receive Initialization" on page 1348 for details on initialization and usage.

Table 37-63. MTA[0-127] – 128 Multicast Table Array Registers

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 5200h at 4h **Offset End:** 5203h at 4h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 5200h at 4h **Offset End:** 5203h at 4h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 5200h at 4h **Offset End:** 5203h at 4h	
Size: 32 bits	**Default:** XXXX_XXXXh				**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	Vector	32b vector of multicast address filter table information.			X	RW

37.6.4.15 RAL[0-15] – Receive Address Low Register

These registers contain the lower bits of the 48 bit Ethernet address.

Table 37-64. RAL[0-15] - Receive Address Low Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 5400h at 8h **Offset End:** 5403h at 8h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 5400h at 8h **Offset End:** 5403h at 8h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 5400h at 8h **Offset End:** 5403h at 8h	
Size: 32 bits	**Default:** XXXXXXXXh				**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RAL	**Receive Address Low.** The lower 32 bits of the 48 bit Ethernet address.			X	RW

37.6.4.16 RAH[0-15] – Receive Address High Register

These registers contain the upper bits of the 48 bit Ethernet address. The complete 48b address is {RAH, RAL}. The first receive address register (RAR0) is also used for exact match pause frame checking (i.e. the destination address matches the first register). Therefore RAR0 should always be used to store the individual Ethernet MAC address of the adapter.

Table 37-65. RAH[0-15] - Receive Address High Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5404h at 8h **Offset End:** 5407h at 8h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5404h at 8h **Offset End:** 5407h at 8h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5404h at 8h **Offset End:** 5407h at 8h	
Size: 32 bits	**Default:** 000XXXXXh			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	AV	**Address valid.** This bit determines whether this address is compared against the incoming packet. Cleared after software reset or Unit Reset. 0 = No match on this address field 1 = Match on this address field		0h	RW
30 : 18	Rsvd	Reserved		0h	RV
17 : 16	ASEL	**Address Select.** Selects how the address is to be used when performing special filtering on receive packets. • 00: Destination address (must be set to this in normal mode) • 01: Source address • 10: Reserved • 11: Reserved		X	RW
15 : 00	RAH	**Receive Address High.** The upper 16 bits of the 48 bit Ethernet address.		X	RW

37.6.4.17 VFTA[0-127] – 128 VLAN Filter Table Array Registers

There is one register per 32 bits of the VLAN Filter Table. The size of the word array depends on the number of bits implemented in the VLAN Filter Table. Software must mask to the desired bit on reads and supply a 32-bit word on writes. The algorithm for VLAN filtering via the VFTA is identical to that used for the Multicast Table Array, refer to "Receive Initialization" on page 1348 for details on initialization and usage.

Table 37-66. VFTA[0-127] - 128 VLAN Filter Table Array Registers

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5600h at 4h **Offset End:** 5603h at 4h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5600h at 4h **Offset End:** 5603h at 4h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5600h at 4h **Offset End:** 5603h at 4h	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	VLAN_Vector	32b vector of VLAN filter table information.		X	RW

37.6.5 Transmit Registers: Detailed Descriptions

37.6.5.1 TCTL – Transmit Control Register

This register controls packet transmission. Packet collision recovery, 64B data padding, and software XOFF transmission are controlled here.

Table 37-67. TCTL: Transmit Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0400h **Offset End:** 0403h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0400h **Offset End:** 0403h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0400h **Offset End:** 0403h	
Size: 32 bits	**Default:** 00000008h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 25	Rsvd	Reserved		0h	RV
24	RTLC	**Re-Transmit on Late Collision.** This bit configures the hardware to perform retransmission of packets when a late collision is detected. Note that the collision window is speed dependent: 64B for 10/100 Mbps and 512B for 1Gbps operation. If a late collision is detected when this bit is clear, the transmit function assumes the packet is successfully transmitted. ***Note:*** This bit is ignored in full-duplex mode.		0h	RW
23	PBE	**Packet Burst Enable.** The EP80579's GbE does not support Packet Bursting for 1Gbps half-duplex transmit operation. This bit must be set to 0.		0h	RV
22	SWXOFF	**Software XOFF Transmission.** When set to a 1 the device will schedule the transmission of an XOFF (PAUSE) frame using the current value of the PAUSE timer. This bit clears itself upon transmission of the XOFF frame. ***Note:*** While 802.3x flow control is only defined during full duplex operation, the sending of PAUSE frames via the SWXOFF bit is not gated by the duplex settings within the device. Software should not write a 1 to this bit while the device is configured for half duplex operation.		0h	RW
21 : 12	COLD	**Collision Distance.** Wire speeds of 1Gbps result in a very short collision radius with traditional minimum packet sizes. This bit specifies the minimum number of bytes in the packet to satisfy the desired collision distance for proper CSMA/CD operation. It is important to note that the resulting packet has special characters appended to the end, not regular data characters. Hardware strips special characters for packets that go from 1 Gbps environments to 100 Mbps environments. ***Note:*** The hardware checks and pads to this value even in full-duplex operation.		0h	RW

Table 37-67. TCTL: Transmit Control Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0400h **Offset End:** 0403h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0400h **Offset End:** 0403h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0400h **Offset End:** 0403h	
Size: 32 bits	**Default:** 00000008h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
11 : 04	CT	**Collision Threshold.** Software may choose to abort packet transmission in less than the Ethernet mandated 16 collisions. This field determines the number of attempts at retransmission prior to giving up on the packet (not including the first transmission attempt). The Ethernet back-off algorithm is implemented and clamps to the maximum number of slot-times after 10 retries. This field only has meaning when in half-duplex operation. ***Note:*** While this field can be varied, it should be set to a value of 15 in order to comply with the IEEE specification requiring a total of 16 attempts.		0h	RW
03	PSP	**Pad Short Packets** to 64B with valid data characters, NOT padding symbols. 0 = Do not pad short packets 1 = Pad short packets ***Note:*** This is not the same as the mini-mum collision distance.		1h	RW
02	Rsvd	Reserved.		0h	RV
01	EN	Enable. 0 = Writing this bit to 0 will stop transmission after any in progress packets are sent. Data remains in the transmit FIFO until the device is re-enabled. Software should combine this with reset if the packets in the FIFO should be flushed. 1 = The transmitter is enabled.		0h	RW
00	Rsvd	Reserved.		0h	RV

37.6.5.2 TIPG – Transmit IPG Register

This register controls the Inter Packet Gap (IPG) timer.

Table 37-68. TIPG: Transmit IPG Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0410h **Offset End:** 0413h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0410h **Offset End:** 0413h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0410h **Offset End:** 0413h	
Size: 32 bits	**Default:** 00602008h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 30	Rsvd	Reserved		0h	RV
29 : 20	IPGR2	**IPG Receive Time 2.** Specifies the total length of the IPG time for non back-to-back transmissions. Measured in increments of the MAC clock: • 8 ns MAC clock when operating @ 1 Gbps (82544GC/EI only). • 80 ns MAC clock when operating @ 100 Mbps • 800 ns MAC clock when operating @ 10 Mbps. In order to calculate the actual IPG value, a value of six should be added to the IPGR2 value as six MAC clocks are used by the MAC for synchronization and internal engines. For the IEEE 802.3 standard IPG value of 96-bit time, the value that should be programmed into IPGR2 is six (total IPG delay of 12 MAC clock cycles) According to the IEEE802.3 standard, IPGR1 should be 2/3 of IPGR2 value.IPGR2 is significant only in half-duplex mode of operation.		0x6h	RW

Table 37-68. TIPG: Transmit IPG Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0410h **Offset End:** 0413h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0410h **Offset End:** 0413h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0410h **Offset End:** 0413h	
Size: 32 bits	**Default:** 00602008h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
19 : 10	IPGR1	**IPG Receive Time 1.** Specifies the length of the first part of the IPG time for non back-to- back transmissions. During this time, the internal IPG counter restarts if any carrier event occurs. Once the time specified in IPGR1 has elapsed, carrier sense does not affect the IPG counter. According to the IEEE802.3 standard, IPGR1 should be 2/3 of IPGR2 value. Measured in increments of the MAC clock: • 8 ns MAC clock when operating @ 1 Gbps • 80 ns MAC clock when operating @ 100 Mbps • 800 ns MAC clock when operating @ 10 Mbps. For IEEE 802.3 minimum IPG value of 96-bit time, the value that should be programmed into IPGR1 is eight. IPGR1 is significant only in half-duplex mode of operation.		0x8h	RW
09 : 00	IPGT	**IPG Transmit Time** Specifies the IPG time for back-to-back packet transmissions Measured in increments of the MAC clock: • 8 ns MAC clock when operating @ 1 Gbps. • 80 ns MAC clock when operating @ 100 Mbps. • 800 ns MAC clock when operating @ 10 Mbps. To calculate the IPG value for 10/100/1000BASE-T applications, a value of four should be added to the IPGT value as four clocks are used by the MAC as internal overhead. The value that should be programmed into IPGT is 8. These values are recommended to assure that the minimum IPG gap is met under all synchronization conditions.		0x8h	RW

37.6.5.3 AIT – Adaptive IFS Throttle Register

Adaptive IFS throttles back-to-back transmissions in the transmit packet buffer and delays their transfer to the CSMA/CD transmit function. This can be used to delay the transmission of back-to-back packets on the wire. Normally, this register should be set to 0. However, if additional delay is desired between back-to-back transmits, then this register may be set with a value greater than zero.

The Adaptive IFS Throttle timer provides a similar function to the "TIPG – Transmit IPG Register" on page 1493, however, Adaptive IFS affects only the initial transmission timing, not re-transmission timing.

Table 37-69. AIT: Adaptive IFS Throttle Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 0458h **Offset End:** 045Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 0458h **Offset End:** 045Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 0458h **Offset End:** 045Bh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved			0h	RV
15 : 00	AIFS	Adaptive IFS Value Adaptive IFS throttles back-to-back transmissions in the transmit packet buffer and delays their transfer to the CSMA/CD transmit function. Normally, this register should be set to 0b. However, if additional delay is desired between back-to-back transmit packets, then this register can be set with a value greater than zero (0). This feature can be helpful in high collision half-duplex environments. In order for AIFS to take effect it should be larger than the minimum IFS value defined in IEEE 802.3 standard. AIFS has no effect on transmissions that occur immediately after receives or transmissions that are not back-to-back. In addition, it has no effect on re-transmission timing (retransmission after collisions). The AIFS value is additive to the TIPG.IPGT value. This time unit for this value is speed dependent: 1000Mbps is 8ns 100Mbps is 80ns 10 Mbps is 800ns			0h	RW

37.6.5.4 TDBAL – Transmit Descriptor Base Address Low Register

This register contains the lower bits of the 64 bit descriptor base address. The Transmit Descriptor Base Address must point to a 16B aligned block of data (i.e. the lower 4 bits are always 0).

Table 37-70. TDBAL: Transmit Descriptor Base Address Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3800h **Offset End:** 3803h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3800h **Offset End:** 3803h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3800h **Offset End:** 3803h	
Size: 32 bits	**Default:** XXXXXXX0h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	TDBAL	Transmit Descriptor Base Address Low		X	RW
03 : 00	0	Writes are ignored, reads return 0.		0h	RV

37.6.5.5 TDBAH – Transmit Descriptor Base Address High Register

This register contains the upper 32 bits of the 64 bit Descriptor base address.

Table 37-71. TDBAH: Transmit Descriptor Base Address High Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3804h **Offset End:** 3807h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3804h **Offset End:** 3807h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3804h **Offset End:** 3807h	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TDBAH	Transmit Descriptor Base Address ***Note:*** TDBAH[31:0] must be set to 0.		X	RW

37.6.5.6 TDLEN – Transmit Descriptor Length Register

This register contains the descriptor length and must be 128B aligned (i.e. the lower 7 bits are always 0).

Table 37-72. TDLEN: Transmit Descriptor Length Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 3808h **Offset End:** 380Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 3808h **Offset End:** 380Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 3808h **Offset End:** 380Bh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	Rsvd	Reserved			0h	RV
19 : 07	LEN	Descriptor Length			0h	RW
06 : 00	0	Writes are ignored, reads return 0.			0h	RV

37.6.5.7 TDH – Transmit Descriptor Head Register

This register contains the head pointer for the transmit descriptor ring. It points to a 16B datum. Hardware controls this pointer. The only time that software should write to this register is after a reset (hardware reset or CTRL.RST) and before enabling the transmit function (TCTL.EN). Writing this register while the transmit function is enabled will cause indeterminate behavior.

Table 37-73. TDH: Transmit Descriptor Head Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 3810h **Offset End:** 3813h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 3810h **Offset End:** 3813h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 3810h **Offset End:** 3813h	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved			0h	RV
15 : 00	TDH	Transmit Descriptor Head			0h	RW

37.6.5.8 TDT – Transmit Descriptor Tail Register

This register contains the tail pointer for the transmit descriptor ring. It points to a 16B datum. Software writes the tail pointer to add more descriptors to the transmit ready queue. Hardware attempts to transmit all packets referenced by descriptors between head and tail.

Table 37-74. TDT: Transmit Descriptor Tail Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3818h **Offset End:** 381Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3818h **Offset End:** 381Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3818h **Offset End:** 381Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Core Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	TDT	Transmit Descriptor Tail		0h	RW

37.6.5.9 TIDV – Transmit Interrupt Delay Value Register

This register is used to delay interrupt notification for transmit operations by coalescing interrupts for multiple transmitted buffers. Delaying interrupt notification helps maximize the amount of transmit buffers reclaimed by a single interrupt. This feature ***only*** applies to transmit descriptor operations where interrupt-based reporting is requested (RS set) and the use of the timer function is requested (IDE is set).

Table 37-75. TIDV: Transmit Interrupt Delay Value Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3820h **Offset End:** 3823h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3820h **Offset End:** 3823h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3820h **Offset End:** 3823h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	IDV	**Interrupt Delay Value.** Timer increments are RMII: 1.28 microseconds RGMII: 1.024 microseconds. • This register is used to delay interrupt notification for transmit operations by coalescing interrupts for multiple transmitted buffers. Delaying interrupt notification helps maximize the amount of transmit buffers reclaimed by a single interrupt. This feature only applies to transmit descriptor operations where (a) interrupt-based reporting is requested (RS set) and (b) the use of the timer function is requested (IDE is set). • This feature operates by initiating a countdown timer upon successfully transmitting the buffer. If a subsequent transmit delayed-interrupt is scheduled before the timer expires, the timer is re-initialized to the programmed value and re-starts its countdown. When the timer expires, a transmit-complete interrupt (ICR.TXDW) is generated. • Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. • Setting the value to 0 is not allowed. If an immediate (non-scheduled) interrupt is desired for any transmit descriptor, the descriptor IDE should be set to 0. • The occurrence of either an immediate (non-scheduled) or absolute transmit timer interrupt will halt the TIDV timer and eliminate any spurious second interrupts. • Transmit interrupts due to a Transmit Absolute Timer (TADV) expiration or an immediate interrupt (RS =1, IDE=0) will cancel a pending TIDV interrupt. The TIDV countdown timer is reloaded but halted, though it may be restarted by a processing a subsequent transmit descriptor.		0h	RW

37.6.5.10 TXDCTL – Transmit Descriptor Control Register

This register controls the fetching and write-back of transmit descriptors. The three threshold values are used to determine when descriptors will be read from and written to host memory. The values may be in units of cache lines or 16B descriptors.

Table 37-76. TXDCTL: Transmit Descriptor Control Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3828h **Offset End:** 382Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3828h **Offset End:** 382Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3828h **Offset End:** 382Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 25	LWTHRESH	**Transmit Descriptor Low Threshold.** This field controls the number of pre-fetched transmit descriptors at which a transmit descriptor-low interrupt is reported. Asserting ICR.TXD_LOW only when the processing distance from the TDT register drops below LWTHRESH may allow software to operate more efficiently by maintaining a continuous addition of transmit work, interrupting only when the hardware nears completion of all submitted work. An interrupt condition is asserted when the number of descriptors available transitions from threshold_level + 1 -> threshold_level where LWTHRESH specifies a multiple of 8 descriptors, (i.e. threshold_level = 8*LWTHRESH). Setting this value to 0 will cause this interrupt to be generated only when the transmit descriptor cache becomes completely empty.		0h	RW
24	GRAN	**Granularity** of the thresholds in this register. 0 = Cache Lines 1 = Descriptors (16B each)		0h	RW
23 : 22	Rsvd	Reserved		0h	RV

Table 37-76. TXDCTL: Transmit Descriptor Control Register (Sheet 2 of 2)

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 3828h **Offset End:** 382Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 3828h **Offset End:** 382Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 3828h **Offset End:** 382Bh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
21 : 16	WTHRESH	**Write-back Threshold.** This field controls the write-back of processed transmit descriptors. This threshold refers to the number of transmit descriptors in the GbE hardware buffer which are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back will occur only after more than WTHRESH descriptors are available for write-back. Since write-back notification of transmit descriptor completion is optional (under the control of the RS bit in the descriptor), not all processed descriptors are counted with respect to WTHRESH (any single transmit descriptor with RS=0 is consumed with no writeback notification performed). When WTHRESH is non-zero, processing a descriptor with RS=1 initiates accumulation of pending writebacks; accumulated writebacks will include even those descriptors with RS=0, in order to optimize writeback bursts. ***Note:*** When WTHRESH value is set to 0, transmit descriptor writeback notification will be similar to the 82452 behavior. In accordance with WTHRESH=0, the writeback notification for a descriptor with RS=1 will occur as soon as the descriptor is processed. In addition, processed transmit descriptors are not written-back in entirety; only the descriptor status field is written back/ updated. This 82542-compatible mode is the default HW behavior.		0h	RW
15 : 14	Rsvd	Reserved			0h	RV
13 : 08	HTHRESH	**Host Threshold.** This field is used to control the fetching of descriptors from host memory. This threshold refers to the number of valid, unprocessed receive descriptors that must exist in host memory before they will be fetched.			0h	RW
07 : 06	Rsvd	Reserved			0h	RV
05 : 00	PTHRESH	**Prefetch Threshold.** This field is used to control when a prefetch of descriptors will be considered. This threshold refers to the number of valid, unprocessed transmit descriptors the chip has in its GbE hardware buffer. If this number drops below PTHRESH, the algorithm will consider pre-fetching descriptors from host memory. This fetch will not happen however unless there are at least HTHRESH valid descriptors in host memory to fetch.			0h	RW

37.6.5.11 TADV – Transmit Absolute Interrupt Delay Value Register

The Transmit Absolute Interrupt Delay Value Register (TADV) may be used to coalesce transmit interrupts, however, it may be necessary to ensure that no completed transmission remains unnoticed for too long an interval in order ensure timely release of transmit buffers. This register may be used to ensure that a transmit interrupt occurs at some predefined interval after a transmit is completed. Like the TIDV, the absolute transmit timer only applies to transmit descriptor operations where interrupt-based reporting is requested (RS set) and the use of the timer function is requested (IDE is set).

Table 37-77. TADV: Transmit Absolute Interrupt Delay Value Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 382Ch **Offset End:** 382Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 382Ch **Offset End:** 382Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 382Ch **Offset End:** 382Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Rsvd	Reserved		0h	RV
15 : 00	IDV	**Interrupt Delay Value.** Timer increments are RMII: 1.28 microseconds RGMII: 1.024 microseconds. The transmit interrupt delay timer (TIDV) can be used to coalesce transmit interrupts. However, it might be necessary to ensure that no completed transmit remains unnoticed for too long an interval in order ensure timely release of transmit buffers. This register can be used to ENSURE that a transmit interrupt occurs at some predefined interval after a transmit is completed. Like the delayed-transmit timer, the absolute transmit timer ONLY applies to transmit descriptor operations where (a) interrupt-based reporting is requested (RS set) and (b) the use of the timer function is requested (IDE is set). This feature operates by initiating a countdown timer upon successfully transmitting the buffer. When the timer expires, a transmit-complete interrupt (ICR.TXDW) is generated. The occurrence of either an immediate (non-scheduled) or delayed transmit timer (TIDV) expiration interrupt halts the TADV timer and eliminates any spurious second interrupts. Setting the value to 0b disables the transmit absolute delay function. If an immediate (nonscheduled) interrupt is desired for any transmit descriptor, the descriptor IDE should be set to 0b. ***Note:*** This timer ONLY causes an interrupt. It does NOT cause a writeback		0h	RW

37.6.5.12 TSPMT – TCP Segmentation Pad and Minimum Threshold Register

This register specifies fields affecting hardware behavior during TCP Segmentation operations. For normal (non TCP Segmentation) operations, the transmit DMA never begins servicing an individual data descriptor unless the transmit Packet Buffer has sufficient room to accept all of the data associated with the descriptor. However, for TCP Segmentation operations, it may be desirable to use a data descriptor which refers to a larger contiguous buffer in host memory than is actually allocated for the transmit Packet Buffer. For this case, the transmit DMA must be able to initiate smaller transfers than the entire descriptor's data length field (i.e., during TCP segmentation, the transmit DMA does not wait until the entire descriptor's data can fit in the packet buffer).

When performing TCP segmentation, the packet prototype header initially transferred by DMA is stored internally and updated as each packet of the TCP segmentation operation is composed. As data for subsequent TCP segments is DMA'd into the controller, the frame header for each segment is dynamically inserted in front of the frame payload data stream prior to being written to the packet buffer. In order to obtain the most efficient use of burst DMA operations, the transmit DMA will attempt to fetch as much data from a descriptor as possible, rather than limiting itself to bursting each data segment individually. However, to do this, sufficient packet-buffer space must be reserved to account for all headers which will be inserted into the fetched data stream, as the burst may span multiple data segments. The calculation of how much packet buffer space should be reserved is dependent on the MSS being used in the packet header, the maximum-sized data buffer pointed to by a descriptor, and the current header size. Such calculation cannot be easily calculated in hardware, and is left to software to pre-calculate for the worst-case usage.

The transmit DMA will further refrain from initiating service of a new data descriptor unless sufficient packet buffer space exists to at least fetch a full data segment or complete a partially-fetched segment. This additionally helps to reduce the number of small DMA bursts and reducing the efficiency of the host interface.

Table 37-78. TSPMT: TCP Segmentation Pad And Minimum Threshold Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 3830h **Offset End:** 3833h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 3830h **Offset End:** 3833h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 3830h **Offset End:** 3833h	
Size: 32 bits	**Default:** 01000400h			**Power Well:**	GbE0: Core Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	TSPBP	**TCP Segmentation Packet Buffer Padding**, value is in bytes. This field allows software configuration of packet buffer space which must be reserved as “pad” for worst-case header insertion. To ensure that this value does not prevent descriptors from being serviced at all, it is necessary that the transmit packet buffer allocation should be larger than the sum of (maximum TCP HDRLEN + maximum MSS + TSPMT.TMPBP + 80 bytes).		0x0100h	RW
15 : 00	TSMT	**TCP Segmentation Minimum Transfer**, value is in bytes. The DMA will attempt to issue burst fetches for as much data as possible, and it is possible for the transmit DMA to cause the transmit packet buffer to approach fullness (less the pad specified). However, if the packet buffer empties slightly, the transmit DMA could initiate a series of small transfers. To further optimize the efficiency of the transmit DMA during TCP segmentation operation, the this TSPMT.TSMT field allows software configuration of the minimum number of bytes which the DMA should attempt to transfer in a single burst operation. The transmit DMA will use this value to refrain from issuing a burst read until at least TSPMT.TSMT bytes of data from the current data descriptor can be stored in the packet buffer. This check will be ignored if, after a series of DMA operations, the descriptor contains a smaller number of unfetched data bytes. To ensure that this minimum threshold does not prevent descriptors from being serviced at all, it is necessary that the transmit packet buffer allocation should be larger than the sum of (TSPMT.TSMT + TSPMT.TSPBP + 80 bytes).		0x0400h	RW

37.6.6 Statistical Registers: Detailed Descriptions

The statistics registers generally assume the presence of a system host driver which has enabled transmit and receive operations between the GbE controller and host memory. Throughout the descriptions of specific statistics registers, many require the driver to have enabled transmits enabled (TCTL.EN=1) or receives enabled (RCTL.EN=1). Some of the registers provide an indication of link activity even when the driver has transmits or receives disabled.

All statistics registers are cleared when read. It is the responsibility of software to maintain incremental count variables if statistics are intended to be monitored regularly for extended periods of time. All registers “stick” at a maximum value of 0xFFFFFFFF when reached (until cleared by reset or a read) - if accurate statistics are required, it is expected that software will query statistics at sufficient intervals to avoid reaching maximum values and failing to count values unexpectedly.

Note: For the receive statistics it should be noted that a packet is indicated as “received” if it passes the device's filters and is directed towards Receive Packet Buffer memory. A packet does not have to be received all the way to the driver in order to be counted as “received”.

Note: Due to divergent paths between interrupt-generation and logging of relevant statistics counts, it may be possible to generate an interrupt to the system for a noteworthy event prior to the associated statistics count actually being incremented. This is extremely unlikely due to expected delays associated with the system interrupt-collection and ISR delay, but might be observed as an interrupt for which statistics values do not quite make sense. Hardware guarantees that any event noteworthy of inclusion in a statistics count will be reflected in the appropriate count within 1 usec; a small time-delay prior to read of statistics may be necessary to avoid the potential for receiving an interrupt and observing an inconsistent statistics count as part of the ISR.

37.6.6.1 CRCERRS – CRC Error Count Register

This register counts the number of receive packets with CRC errors. In order for a packet to be counted in this register, it must pass MAC address filtering (Broadcast or Individual-Address/Multicast match) and must be 64B or greater (from <Destination Address> through <CRC>, inclusively) in length.

Table 37-79. CRCERRS: CRC Error Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4000h **Offset End:** 4003h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4000h **Offset End:** 4003h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4000h **Offset End:** 4003h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CRCERRS	CRC error count		0h	RC

37.6.6.2 ALGNERRC – Alignment Error Count Register

This register counts the number of receive packets with alignment errors (i.e. the packet is not an integer number of bytes in length). In order for a packet to be counted in this register, it must pass MAC address filtering (Broadcast or Individual-Address/ Multicast match) and must be 64B or greater (from <Destination Address> through <CRC>, inclusively) in length. This register is valid only in GMII/MII mode, during 10/ 100 Mbps operation.

Table 37-80. ALGNERRC: Alignment Error Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4004h **Offset End:** 4007h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4004h **Offset End:** 4007h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4004h **Offset End:** 4007h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ALGNERRC	Alignment error count		0h	RC

37.6.6.3 RXERRC – Receive Error Count Register

This register counts the number of packets received in which RX_ER was asserted by the PHY. In order for a packet to be counted in this register, it must pass MAC address filtering (Broadcast or Individual-Address/Multicast match) and must be 64Bor greater (from <Destination Address> through <CRC>, inclusively) in length.

Table 37-81. RXERRC: Receive Error Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 400Ch **Offset End:** 400Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 400Ch **Offset End:** 400Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 400Ch **Offset End:** 400Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RXERRC	RX Error Count		0h	RC

37.6.6.4 MPC – Missed Packet Count Register

This register counts the number of missed packets. Packets are missed when the receive FIFO has insufficient space to store the incoming packet. This could be caused because of too few buffers allocated, or because there is insufficient bandwidth on the internal bus. Events setting this counter cause the receiver overrun interrupt condition (ICR.RXO) to be set. This register does not count packets dropped due to the receiver being disabled.

Note: Missed packets will be included/counted in the "TPR – Total Packets Received Register" on page 1528 as well as in the Total Octets Received counter of the "TORL – Total Octets Received Low Register" on page 1525 and the "TORH – Total Octets Received High Register" on page 1526.

Table 37-82. MPC: Missed Packet Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 4010h **Offset End:** 4013h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 4010h **Offset End:** 4013h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 4010h **Offset End:** 4013h	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	MPC	Missed Packets Count			0h	RC

37.6.6.5 SCC – Single Collision Count Register

This register counts the number of times that a successfully transmitted packet encountered a single collision. This register will only increment if transmits are enabled and the device is in half-duplex mode.

Table 37-83. SCC: Single Collision Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 4014h **Offset End:** 4017h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 4014h **Offset End:** 4017h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 4014h **Offset End:** 4017h	
Size: 32 bits	**Default:** 0000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	SCC	Number of times a transmit encountered a single collision.			0h	RC

37.6.6.6 ECOL – Excessive Collisions Count Register

When 16 or more collisions have occurred on a packet, this register increments, regardless of the value of collision threshold. If collision threshold is set below 16, this counter won't increment. This register will only increment if transmits are enabled and the device is in half-duplex mode.

Table 37-84. ECOL: Excessive Collisions Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4018h **Offset End:** 401Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4018h **Offset End:** 401Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4018h **Offset End:** 401Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ECOL	Number of packets with more than 16 collisions		0h	RC

37.6.6.7 MCC – Multiple Collision Count Register

This register counts the number of times that a transmit encountered more than one collision but less than 16. This register will only increment if transmits are enabled and the device is in half-duplex mode.

Table 37-85. MCC: Multiple Collision Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 401Ch **Offset End:** 401Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 401Ch **Offset End:** 401Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 401Ch **Offset End:** 401Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	MCC	Number of times a successful transmit encountered multiple collisions.		0h	RC

37.6.6.8 LATECOL – Late Collisions Count Register

Late collisions are collisions that occur after one slot time. This register will only increment if transmits are enabled and the device is in half-duplex mode.

Table 37-86. LATECOL: Late Collisions Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4020h **Offset End:** 4023h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4020h **Offset End:** 4023h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4020h **Offset End:** 4023h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	LATECOL	Number of packets with late collisions		0h	RC

37.6.6.9 COLC – Collision Count Register

This register counts the total number of collisions seen by the transmitter. This register will only increment if transmits are enabled and the device is in half-duplex mode. This register applies to clear as well as secure traffic.

Table 37-87. COLC: Collision Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4028h **Offset End:** 402Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4028h **Offset End:** 402Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4028h **Offset End:** 402Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	COLC	Total number of collisions experienced by the transmitter		0h	RC

37.6.6.10 DC – Defer Count Register

This register counts defer events. A defer event occurs when the transmitter cannot immediately send a packet due to the medium busy either because another device is transmitting, half-duplex deferral events, or reception of XOFF frames. This register will only increment if transmits are enabled.

Table 37-88. DC: Defer Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4030h **Offset End:** 4033h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4030h **Offset End:** 4033h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4030h **Offset End:** 4033h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	DC	Number of defer events.		0h	RC

37.6.6.11 TNCRS – Transmit with No CRS Count Register

This register counts the number of successful packet transmission in which the CRS signal from the PHY was not asserted within one slot time of start of transmission from the MAC. Start of transmission is defined as the assertion of TX_EN to the PHY. The PHY should assert CRS during every transmission. Failure to do so may indicate that the link has failed, or the PHY has an incorrect link configuration. This register will only increment if transmits are enabled and is only valid when the device is operating at half duplex.

Table 37-89. TNCRS: Transmit with No CRS Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4034h **Offset End:** 4037h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4034h **Offset End:** 4037h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4034h **Offset End:** 4037h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TNCRS	Number of transmissions without a CRS assertion from the PHY.		0h	RC

37.6.6.12 CEXTERR – Carrier Extension Error Count Register

This register counts the number of packets received in which a carrier extension error was signaled by the PHY (by the encoding of 0x1F on the receive data inputs while RX_ER is asserted to the MAC) during the carrier extended time of a packet reception. This register will only increment when the driver has receives enabled and the device is operating at 1000Mb/s.

This counter is non-functional as the receiver doesn't detect Carrier-Extend errors.

Table 37-90. CEXTERR: Carrier Extension Error Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 403Ch **Offset End:** 403Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 403Ch **Offset End:** 403Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 403Ch **Offset End:** 403Fh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CEXTERR	Number of packets received with a carrier extension error.			0h	RC

37.6.6.13 RLEC – Receive Length Error Count Register

This register counts Receive Length Error events. A length error occurs if an incoming packet passes the MAC address filtering (Broadcast or Individual-Address/Multicast match) but is undersized or oversized. Packets less than 64B are deemed as undersized; packets over 1522B are deemed oversized if RCTL.LPE=0. If RCTL.LPE=1, then an incoming packet is only considered oversized if it exceeds 16384B.

Table 37-91. RLEC: Receive Length Error Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 4040h **Offset End:** 4043h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 4040h **Offset End:** 4043h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 4040h **Offset End:** 4043h	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RLEC	Number of packets with receive length errors.			0h	RC

37.6.6.14 XONRXC – XON Received Count Register

This register counts the number of XON packets received. XON packets can use the global address or the station address. This register will only increment if the driver has receives enabled.

Table 37-92. XONRXC: XON Received Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 4048h **Offset End:** 404Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 4048h **Offset End:** 404Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 4048h **Offset End:** 404Bh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	XONRXC	Number of XON packets received.			0h	RC

37.6.6.15 XONTXC – XON Transmitted Count Register

This register counts the number of XON packets transmitted. These packets can be either hardware-initiated due to queue room availability or due to software-initiated action (using TCTL.SWXOFF). This register will only increment if transmits are enabled.

Table 37-93. XONTXC: XON Transmitted Count Register

Description:						
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0		**Offset Start:** 404Ch **Offset End:** 404Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0		**Offset Start:** 404Ch **Offset End:** 404Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0		**Offset Start:** 404Ch **Offset End:** 404Fh	
Size: 32 bits	**Default:** 00000000h				**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**		**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	XONTXC	Number of XON packets transmitted.			0h	RC

37.6.6.16 XOFFRXC – XOFF Received Count Register

This register counts the number of XOFF packets received. XOFF packets can use the global address or the station address. This register will only increment if the driver has receives enabled.

Table 37-94. XOFFRXC: XOFF Received Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4050h **Offset End:** 4053h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4050h **Offset End:** 4053h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4050h **Offset End:** 4053h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	XOFFRXC	Number of XOFF packets received.		0h	RC

37.6.6.17 XOFFTXC – XOFF Transmitted Count Register

This register counts the number of XOFF packets transmitted. These packets can be either hardware-initiated due to queue fullness, or due to software-initiated action (using TCTL.SWXOFF). This register will only increment if transmits are enabled.

Table 37-95. XOFFTXC: XOFF Transmitted Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4054h **Offset End:** 4057h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4054h **Offset End:** 4057h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4054h **Offset End:** 4057h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	XOFFTXC	Number of XOFF packets transmitted.		0h	RC

37.6.6.18 FCRUC – FC Received Unsupported Count Register

This register counts the number of unsupported flow control frames that are received. This counter is incremented when a packet is received which matches either the reserved flow control multicast address (in FCAH/FCAL) or the MAC station address, has a matching flow control type field match (to the value in FCT), but has an incorrect opcode field. This register will only increment if the driver has receives enabled.

Table 37-96. FCRUC: FC Received Unsupported Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4058h **Offset End:** 405Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4058h **Offset End:** 405Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4058h **Offset End:** 405Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	FCRUC	Number of unsupported flow control frames received		0h	RC

37.6.6.19 PRC64 – Good Packets Received Count (64 Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are exactly 64B (from <Destination Address> through <CRC>, inclusively) in length. Hardware flow-control packets are not included in this count. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507).

Table 37-97. PRC64: Good Packets Received Count (64 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 405Ch **Offset End:** 405Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 405Ch **Offset End:** 405Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 405Ch **Offset End:** 405Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC64	Number of good packets received exactly 64 bytes in length.		0h	RC

37.6.6.20 PRC127 – Good Packets Received Count (65-127 Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are from 65B-127B (from <Destination Address> through <CRC>, inclusively) in length. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507).

Table 37-98. PRC127: Good Packets Received Count (65-127 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4060h **Offset End:** 4063h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4060h **Offset End:** 4063h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4060h **Offset End:** 4063h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC127	Number of good packets received, (65-127) bytes in length		0h	RC

37.6.6.21 PRC255 – Good Packets Received Count (128-255 Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are from 128B-255B (from <Destination Address> through <CRC>, inclusively) in length. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507).

Table 37-99. PRC255: Good Packets Received Count (128-255 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4064h **Offset End:** 4067h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4064h **Offset End:** 4067h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4064h **Offset End:** 4067h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC255	Number of good packets received, (128-255) bytes in length.		0h	RC

37.6.6.22 PRC511 – Good Packets Received Count (256-511 Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are from 256B-511B (from <Destination Address> through <CRC>, inclusively) in length. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507).

Table 37-100.PRC511 - Good Packets Received Count (256-511 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4068h **Offset End:** 406Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4068h **Offset End:** 406Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4068h **Offset End:** 406Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC511	Number of good packets received, (256-511) bytes in length		0h	RC

37.6.6.23 PRC1023 – Good Packets Received Count (512-1023 Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are from 512B-1023B (from <Destination Address> through <CRC>, inclusively) in length. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507).

Table 37-101.PRC1023: Good Packets Received Count (512-1023 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 406Ch **Offset End:** 406Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 406Ch **Offset End:** 406Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 406Ch **Offset End:** 406Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC1023	Number of good packets received, (512-1023) bytes in length		0h	RC

37.6.6.24 PRC1522 – Good Packets Received Count (1024 to Max Bytes) Register

This register counts the number of good packets (no link or CRC error) received that are from 1024B-(Max) bytes (from <Destination Address> through <CRC>, inclusively) in length. The value (Max) is dependent on the current receiver configuration (i.e., RCTL.LPE, etc.) and the type of packet being received. This register includes good regular packets received to the Receive Packet Buffer. Packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507). Regular packets identified as Oversized Packets (larger than Max) are also not included in this count (refer to the "ROC – Receive Oversize Count Register" on page 1524).

Note: Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, the nominal value for (Max) is 1522B length.

Table 37-102.PRC1522: Good Packets Received Count (1024 to Max Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4070h **Offset End:** 4073h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4070h **Offset End:** 4073h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4070h **Offset End:** 4073h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PRC1522	Number of good packets received, (1024-Max) bytes in length		0h	RC

37.6.6.25 GPRC – Good Packets Received Count (Total) Register

This register counts the number of good packets (no link or CRC error) received that are of any legal length. This register includes good regular packets received to the Receive Packet Buffer. Regular packets identified as Missed Packets due to Receive Packet Buffer overruns are not included in this count (refer to the "MPC – Missed Packet Count Register" on page 1507). This register should represent the sum of the Good Packets Received (Size=N) counts, and when the driver is enabled, should normally represent the total number of packets received to the driver.

Table 37-103.GPRC: Good Packets Received Count (Total) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4074h **Offset End:** 4077h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4074h **Offset End:** 4077h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4074h **Offset End:** 4077h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GPRC	Number of good packets received (total of all lengths)		0h	RC

37.6.6.26 BPRC – Broadcast Packets Received Count Register

This register counts the number of good (non-erred) broadcast packets received. This register does not count broadcast packets received when the broadcast address filter is disabled.

Table 37-104.BPRC: Broadcast Packets Received Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4078h **Offset End:** 407Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4078h **Offset End:** 407Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4078h **Offset End:** 407Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	BPRC	Number of broadcast packets received		0h	RC

37.6.6.27 MPRC – Multicast Packets Received Count Register

This register counts the number of good (non-erred) multicast packets received. This register does not count multicast packets received that fail to pass the multicast address filtering, nor does it count received flow control packets. This register does not count packets counted by the "MPC – Missed Packet Count Register" on page 1507.

Table 37-105.MPRC: Multicast Packets Received Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 407Ch **Offset End:** 407Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 407Ch **Offset End:** 407Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 407Ch **Offset End:** 407Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	MPRC	Number of multicast packets received		0h	RC

37.6.6.28 GPTC – Good Packets Transmitted Count Register

This register counts the number of good (valid) packets transmitted without error (collision, etc.). A good packet is considered one that is 64B or more in length (from <Destination Address> through <CRC>, inclusively). This count does not include flow control packets transmitted. This register will only increment when transmits are enabled by the driver.

Table 37-106.GPTC: Good Packets Transmitted Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4080h **Offset End:** 4083h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4080h **Offset End:** 4083h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4080h **Offset End:** 4083h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GPTC	Number of good packets transmitted		0h	RC

37.6.6.29 GORCL – Good Octets Received Count Low Register

This is the low 32b of a register that counts the number of octets received which are associated with good packets (no link or CRC errors, no flow-control packets). This counter is incremented for each good regular packet received to the Receive Packet Buffer. Regular packets dropped due to Receive Packet Buffer overruns or due to the driver's receiver being disabled are not included in this count. {GORCH,GORCL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (GORCH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-107.GORCL: Good Octets Received Count Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 4088h **Offset End:** 408Ah	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 4088h **Offset End:** 408Ah	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 4088h **Offset End:** 408Ah	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GORCL	Number of good octets received - lower 4 bytes		0h	RC

37.6.6.30 GORCH – Good Octets Received Count High Register

This is the high 32b of a register that counts the number of octets received which are associated with good packets (no link or CRC errors, no flow-control packets). This counter is incremented for each good regular packet received to the Receive Packet Buffer. Regular packets dropped due to Receive Packet Buffer overruns or due to the driver's receiver being disabled are not included in this count. {GORCH,GORCL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (GORCH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-108.GORCH: Good Octets Received Count High Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 408Ch **Offset End:** 408Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 408Ch **Offset End:** 408Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 408Ch **Offset End:** 408Fh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GORCH	Number of good octets received - upper 4 bytes		0h	RC

37.6.6.31 GOTCL – Good Octets Transmitted Count Low Register

This is the low 32b of a register that counts the number of octets transmitted as good packets. Good packets are considered those which are 64B or more in length (from <Destination Address> through <CRC>, inclusively), and do not encounter an error (collision, etc.) during transmission. This count does not include octets transmitted in flow control packets. This register will only increment when transmits are enabled by the driver. Octets from the <Destination Address> field through the <CRC> field are included in this count. {GOTCH,GOTCL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (GOTCH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-109.GOTCL: Good Octets Transmitted Count Low Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 4090h Offset End: 4093h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 4090h Offset End: 4093h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 4090h Offset End: 4093h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GOTCL	Number of good octets transmitted - lower 4 bytes		0h	RC

37.6.6.32 GOTCH – Good Octets Transmitted Count High Register

This is the high 32b of a register that counts the number of octets transmitted as good packets. Good packets are considered those which are 64B or more in length (from <Destination Address> through <CRC>, inclusively), and do not encounter an error (collision, etc.) during transmission. This count does not include octets transmitted in flow control packets. This register will only increment when transmits are enabled by the driver. Octets from the <Destination Address> field through the <CRC> field are included in this count. {GOTCH,GOTCL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (GOTCH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-110.GOTCH: Good Octets Transmitted Count High Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 4094h Offset End: 4097h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 4094h Offset End: 4097h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 4094h Offset End: 4097h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	GOTCH	Number of good octets transmitted - upper 4 bytes		0h	RC

37.6.6.33 RNBC – Receive No Buffers Count Register

This register counts the number of times that frames were received when there were no available buffers in host memory to store those frames (receive descriptor head and tail pointers were equal). The packet will still be received if there is space in the FIFO. This register does not increment when flow control packets are received. This register will only increment if receives are enabled.

Table 37-111.RNBC: Receive No Buffers Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40A0h **Offset End:** 40A3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40A0h **Offset End:** 40A3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40A0h **Offset End:** 40A3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RNBC	Number of receive no buffer conditions		0h	RC

37.6.6.34 RUC – Receive Undersize Count Register

This register counts the number of received frames that passed address filtering, and were less than minimum size (64B from <Destination Address> through <CRC>, inclusively), and had a valid CRC. This register will only increment if receives are enabled.

Table 37-112.RUC: Receive Undersize Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40A4h **Offset End:** 40A7h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40A4h **Offset End:** 40A7h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40A4h **Offset End:** 40A7h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RUC	Number of receive undersize errors		0h	RC

37.6.6.35 RFC – Receive Fragment Count Register

This register counts the number of received frames that passed address filtering, and were less than minimum size (64B from <Destination Address> through <CRC>, inclusively), but had a bad CRC (this is slightly different from the "RUC – Receive Undersize Count Register" on page 1523). This register will only increment if receives are enabled.

Table 37-113.RFC: Receive Fragment Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:0:0		**Offset Start:** 40A8h **Offset End:** 40ABh	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:1:0		**Offset Start:** 40A8h **Offset End:** 40ABh	
View: PCI 3	**BAR:** CSRBAR	**Bus:Device:Function:** M:2:0		**Offset Start:** 40A8h **Offset End:** 40ABh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RFC	Number of receive fragment errors		0h	RC

37.6.6.36 ROC – Receive Oversize Count Register

This register counts the number of received frames that passed address filtering, and were greater than maximum size. Packets over 1522 bytes are oversized if LongPacketEnable is clear (i.e. RCTL.LPE=0). If LongPacketEnable is set, then an incoming, packet is considered oversized if it exceeds 16384 bytes. If receives are not enabled, this register will not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusively.

Table 37-114.ROC: Receive Oversize Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:0:0		**Offset Start:** 40ACh **Offset End:** 40AFh	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:1:0		**Offset Start:** 40ACh **Offset End:** 40AFh	
View: PCI 3	**BAR:** CSRBAR	**Bus:Device:Function:** M:2:0		**Offset Start:** 40ACh **Offset End:** 40AFh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	ROC	Number of receive oversize errors		0h	RC

37.6.6.37 RJC – Receive Jabber Count Register

This register counts the number of received frames that passed address filtering, and were greater than maximum size and had a bad CRC (this is slightly different from the "ROC – Receive Oversize Count Register" on page 1524). Packets over 1522B are oversized if LongPacketEnable is 0. If LongPacketEnable (LPE) is 1, then an incoming packet is considered oversized if it exceeds 16384 bytes. If receives are not enabled, this register will not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusively.

Table 37-115.RJC: Receive Jabber Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40B0h **Offset End:** 40B3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40B0h **Offset End:** 40B3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40B0h **Offset End:** 40B3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	RJC	Number of receive jabber errors		0h	RC

37.6.6.38 TORL – Total Octets Received Low Register

This is the low 32b of a register that counts the total number of octets received, including octets from both good and bad packets. Packets must first pass the MAC address filtering (Broadcast or Individual-Address/Multicast match) - packets not for this MAC are not included. Octets will be counted for all packets regardless of their length, whether they encountered errors, whether they are detected as regular packets or flow control packets, and regardless of whether they are stored successfully versus dropped from a Receive FIFO (Receive Packet Buffer). Bytes from the <Destination Address> field through the <CRC> field (inclusively) are counted. {TORH,TORL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (TORH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-116.TORL: Total Octets Received Low Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 40C0h Offset End: 40C3h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 40C0h Offset End: 40C3h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 40C0h Offset End: 40C3h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TORL	Number of total octets received - lower 4 bytes		0h	RC

37.6.6.39 TORH – Total Octets Received High Register

This is the high 32b of a register that counts the total number of octets received, including octets from both good and bad packets. Packets must first pass the MAC address filtering (Broadcast or Individual-Address/Multicast match) - packets not for this MAC are not included. Octets will be counted for all packets regardless of their length, whether they encountered errors, whether they are detected as regular packets or flow control packets, and regardless of whether they are stored successfully versus dropped from a Receive FIFO (Receive Packet Buffer). Bytes from the <Destination Address> field through the <CRC> field (inclusively) are counted. {TORH,TORL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (TORH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-117.TORH: Total Octets Received High Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 40C4h Offset End: 40C7h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 40C4h Offset End: 40C7h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 40C4h Offset End: 40C7h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TORH	Number of total octets received - upper 4 bytes		0h	RC

37.6.6.40 TOTL – Total Octets Transmitted Low Register

This the low 32b of a 64b register that counts the total number of octets in successfully transmitted packets. Octets transmitted as part of partial packet transmission (e.g. collisions in half-duplex mode) are not included. Octets will be counted for all packets regardless of their length and regardless of whether regular packets or flow-control packets. Octets from the <Destination Address> field through the <CRC> field are included in this count. {TOTH,TOTL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (TOTH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-118.TOTL: Total Octets Transmitted Low Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40C8h **Offset End:** 40CFh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40C8h **Offset End:** 40CFh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40C8h **Offset End:** 40CFh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TOTL	Number of total octets transmitted - lower 4 bytes		0h	RC

37.6.6.41 TOTH – Total Octets Transmitted High Register

This the high 32b of a 64b register that counts the total number of octets in successfully transmitted packets. Octets transmitted as part of partial packet transmission (e.g. collisions in half-duplex mode) are not included. Octets will be counted for all packets regardless of their length and regardless of whether regular packets or flow-control packets. Octets from the <Destination Address> field through the <CRC> field are included in this count. {TOTH,TOTL} together make up a logical 64-bit register. Each half must be accessed independently using separate 32-bit accesses. Both registers are reset when the upper 32-bit value (TOTH) is read. The register sticks at 0xFFFF_FFFF_FFFF_FFFF.

Table 37-119.TOTH: Total Octets Transmitted High Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 40CCh Offset End: 40CFh	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 40CCh Offset End: 40CFh	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 40CCh Offset End: 40CFh	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TOTH	Number of total octets transmitted - upper 4 bytes		0h	RC

37.6.6.42 TPR – Total Packets Received Register

This register counts the total number of all packets received by the node (must pass the MAC address filtering - Broadcast or Individual-Address/Multicast match). All packets received by the MAC will be counted in this register, regardless of their length, whether they encountered errors, and regardless of whether they are detected as regular packets or flow control packets.

Table 37-120.TPR: Total Packets Received Register

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 40D0h Offset End: 40D3h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 40D0h Offset End: 40D3h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 40D0h Offset End: 40D3h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TPR	Total of all packets received		0h	RC

37.6.6.43 TPT – Total Packets Transmitted Register

These registers count the total number of successfully transmitted packets. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count. All packets will be counted regardless of their length and regardless of whether regular packets or flow-control packets.

Table 37-121.TPT: Total Packets Transmitted Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40D4h **Offset End:** 40D7h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40D4h **Offset End:** 40D7h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40D4h **Offset End:** 40D7h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TPT	Number of all packets transmitted		0h	RC

37.6.6.44 PTC64 – Packets Transmitted Count (64 Bytes) Register

This register counts the number of packets successfully transmitted that are exactly 64B (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count. Packet transmissions shorter than 64 bytes which may be generated due to having Short-Frame Padding (PSP) disabled are not included. This register also does not include any flow control packets transmitted.

Table 37-122.PTC64 - Packets Transmitted Count (64 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40D8h **Offset End:** 40DBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40D8h **Offset End:** 40DBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40D8h **Offset End:** 40DBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PTC64	Number of all packets transmitted that are 64 bytes in length		0h	RC

37.6.6.45 PTC127 – Packets Transmitted Count (65-127 Bytes) Register

This register counts the number of packets successfully transmitted that are 65B-127B (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count.

37.6.6.46 PTC255 – Packets Transmitted Count (128-255 Bytes) Register

This register counts the number of packets successfully transmitted that are 128B-255B (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count.

Table 37-123.PTC255: Packets Transmitted Count (128-255 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40E0h **Offset End:** 40E3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40E0h **Offset End:** 40E3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40E0h **Offset End:** 40E3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PTC255	Number of packets transmitted that are 128-255 bytes in length		0h	RC

37.6.6.47 PTC511 – Packets Transmitted Count (256-511 Bytes) Register

This register counts the number of packets successfully transmitted that are 256B-511B (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count.

Table 37-124.PTC511: Packets Transmitted Count (256-511 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40E4h **Offset End:** 40E7h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40E4h **Offset End:** 40E7h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40E4h **Offset End:** 40E7h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PTC511	Number of packets transmitted that are 256-511 bytes in length		0h	RC

37.6.6.48 PTC1023 – Packets Transmitted Count (512-1023 Bytes) Register

This register counts the number of packets successfully transmitted that are 512B-1023B (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g. collisions in half-duplex mode) are not included in this count.

Table 37-125.PTC1023: Packets Transmitted Count (512-1023 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40E8h **Offset End:** 40EBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40E8h **Offset End:** 40EBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40E8h **Offset End:** 40EBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PTC1023	Number of packets transmitted that are 512-1023 bytes in length		0h	RC

37.6.6.49 PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register

This register counts the number of packets successfully transmitted that are 1024B or longer (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (e.g., collisions in half-duplex mode) are not included in this count.

Table 37-126.PTC1522: Packets Transmitted Count (1024-1522 Bytes) Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40ECh **Offset End:** 40EFh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40ECh **Offset End:** 40EFh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40ECh **Offset End:** 40EFh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	PTC1522	Number of packets transmitted that are 1024 or more bytes in length		0h	RC

37.6.6.50 MPTC – Multicast Packets Transmitted Count Register

This register counts the number of multicast packets transmitted. This register does not include flow control packets and increments only if transmits are enabled.

Table 37-127.MPTC: Multicast Packets Transmitted Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40F0h **Offset End:** 40F3h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40F0h **Offset End:** 40F3h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40F0h **Offset End:** 40F3h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	MPTC	Number of multicast packets transmitted		0h	RC

37.6.6.51 BPTC – Broadcast Packets Transmitted Count Register

This register counts the number of broadcast packets transmitted. This register will only increment if transmits are enabled.

Table 37-128.BPTC: Broadcast Packets Transmitted Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40F4h **Offset End:** 40F7h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40F4h **Offset End:** 40F7h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40F4h **Offset End:** 40F7h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	BPTC	Number of broadcast packets transmitted count		0h	RC

37.6.6.52 TSCTC – TCP Segmentation Context Transmitted Count Register

This register counts the number of TCP segmentation off load transmissions and increments once the last portion of the TCP segmentation context payload is segmented and loaded as a packet into the GbE hardware transmit buffer. Note that it is not a measurement of the number of packets sent out (covered by other registers). This register will only increment if transmits and TCP Segmentation off load are enabled.

Table 37-129.TSCTC: TCP Segmentation Context Transmitted Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40F8h **Offset End:** 40FBh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40F8h **Offset End:** 40FBh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40F8h **Offset End:** 40FBh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TSCTC	Number of TCP Segmentation contexts transmitted count		0h	RC

37.6.6.53 TSCTFC – TCP Segmentation Context Transmit Fail Count Register

This register counts the number of TCP segmentation off load requests to the hardware that failed to transmit all data in the TCP segmentation context payload (due to the context specifying less data than the data descriptors reference). TCP Segmentation requires the context's length specification to match the total length of the supplied data descriptors, however. Therefore, this count should always be zero if TCP Segmentation is properly used.

Table 37-130.TSCTFC: TCP Segmentation Context Transmit Fail Count Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 40FCh **Offset End:** 40FFh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 40FCh **Offset End:** 40FFh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 40FCh **Offset End:** 40FFh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	TSCTFC	Number of TCP Segmentation contexts where the device failed to transmit the entire data payload		0h	RC

37.6.7 Management Register Descriptions

37.6.7.1 WUC – Wake Up Control Register (0x05800; RW)

Table 37-131.WUC - Wake Up Control Register (0x05800; RW)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5800h **Offset End:** 5803h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5800h **Offset End:** 5803h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5800h **Offset End:** 5803h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	RSVD	Reserved		0h	RO
03	APMPME	**Assert PME On APM Wakeup –** If it is 1, the GbE will set the PME_Status bit in the Power Management Control / Status Register (PMCSR) and assert GBE_PME_WAKE when APM Wakeup is enabled and the GbE receives a matching magic packet. *Note that this bit is loaded from the EEPROM, if present		0h	RW
02	PME_Status	PME_Status This bit is set when the GbE receives a wakeup event. It is the same as the PME_Status bit in the Power Management Control / Status Register (PMCSR). Writing a “1” to this bit will clear it and clear the PME_Status bit in the PMCSR.		0h	RWC
01	PME_EN	PME_En This read/write bit is used by the driver to access the PME_En bit of the Power Management Control / Status Register (PMCSR) without writing to PCI configuration space.		0h	RW
00	APME	**Advance Power Management Enable** - If “1”, APM Wakeup is enabled. *Note that this bit is loaded from the EEPROM, if present		0h	RW

Note: The PME_En and PME_Status bits are reset when PWR_GOOD is 0. When AUX_PWR_PRESENT=0, this register is also reset by the deassertion (rising edge) of RESET_N and the transition from D3 to D0. The other bits are reset on the standard internal resets.

37.6.7.2 WUFC – Wake Up Filter Control Register (0x05808; RW)

This register is used to enable/disable each of the pre-defined and flexible filters affecting wake up support. For most bits, a value of 1 means a wakeup for the specific event is enabled, and a value of 0 means the wakeup is disabled. For example, when MC=1, a packet containing an IA which matches the Directed Multicast Filter will generate a wakeup event.

Table 37-132.WUFC - Wake Up Filter Control Register (0x05808; RW)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5808h **Offset End:** 580Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5808h **Offset End:** 580Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5808h **Offset End:** 580Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	RSVD	Reserved. Should be set to 0.		0h	RV
19	FLX3	Flexible Filter 3 Enable		0h	RW
18	FLX2	Flexible Filter 2 Enable		0h	RW
17	FLX1	Flexible Filter 1 Enable		0h	RW
16	FLX0	Flexible Filter 0 Enable		0h	RW
15	RSVD	Reserved. Should be set to 0.		0h	RW
14 : 08	RSVD	Reserved. Should be set to 0.		0h	RV
07	IPV6	Directed IPv6 Packet Wake Up Enable		0h	RW
06	IPV4	Directed IPv4 Packet Wake Up Enable		0h	RW
05	ARP	ARP/IPv4 Request Packet Wake Up Enable		0h	RW
04	BC	Broadcast Wake Up Enable		0h	RW
03	MC	Directed Multicast Wake Up Enable		0h	RW
02	EX	Directed Exact Wake Up Enable		0h	RW
01	MAG	Magic Packet Wake Up Enable		0h	RW
00	Rsvd	Reserved		0h	RV

37.6.7.3 WUS – Wake Up Status Register (0x05810; RW)

This register is used to record statistics about all Wake Up packets received. If a packet matches multiple criteria than multiple bits could be set. Writing a 1 to any bit will clear that bit.

This register will not be cleared when RESET_N is asserted. It will only be cleared when PWR_OK is deasserted or when cleared by the driver.

Prior to re-entering a wakeup-enabled sleep state, this register should be explicitly cleared by writing with all 1's.

Table 37-133.WUS - Wake Up Status Register (0x05810; RW)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5810h **Offset End:** 5813h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5810h **Offset End:** 5813h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5810h **Offset End:** 5813h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 20	RSVD	Reserved. Should be set to 0.		0h	RV
19	FLX3	Flexible Filter 3 Match		0h	RWC
18	FLX2	Flexible Filter 2 Match		0h	RWC
17	FLX1	Flexible Filter 1 Match		0h	RWC
16	FLX0	Flexible Filter 0 Match		0h	RWC
15 : 08	Reserved	Reserved.		0h	RV
07	IPV6	Directed IPv6 Packet Wake Up Packet Received		0h	RWC
06	IPV4	Directed IPv4 Packet Wake Up Packet Received		0h	RWC
05	ARP	ARP/IPv4 Request Packet Wake Up Packet Received		0h	RWC
04	BC	Broadcast Wake Up Packet Received		0h	RWC
03	MC	Directed Multicast Wake Up Packet Received The packet was a multicast packet whose hashed to a value that corresponded to a 1 bit in the Multicast Table Array ***Note:*** If the MAC has been configured for promiscuous mode, a multicast wakeup will occur if a broadcast packet is received. This is because a broadcast message is a special type of multicast message. Refer to 802.3.		0h	RWC
02	EX	Directed Exact Wake Up Packet Received The packet's address matched one of the 16 pre-programmed exact values in the Receive Address registers		0h	RWC
01	MAG	Magic Packet Wake Up Packet Received		0h	RWC
00	Rsvd	Reserved. Must be written as '0'		0h	RV

37.6.7.4 IPAV – IP Address Valid Register (0x05838; RW)

The IP Address Valid indicates whether the IP addresses in the IP Address Table are valid:

Table 37-134.IPAV - IP Address Valid Register (0x05838; RW)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5838h **Offset End:** 583Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5838h **Offset End:** 583Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5838h **Offset End:** 583Bh	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 17	RSVD	Reserved. Should be set to 0.		0h	RV
16	V60	IPv6 Address 0 Valid		0h	RW
15 : 04	RSVD	Reserved. Should be set to 0.		0h	RV
03	V43	IPv4 Address 3 Valid		0h	RW
02	V42	IPv4 Address 2 Valid		0h	RW
01	V41	IPv4 Address 1 Valid		0h	RW
00	V40	IPv4 Address 0 Valid The initial value is loaded from the IP Address Valid bit of the EEPROM's Management Control Register		0h	RW

37.6.7.5 IP4AT[0-3] - (0x5840 - 0x5858; RW) – IPv4 Address Table Registers

The IPv4 Address Table is used to store the four IPv4 addresses for ARP/IPv4 Request packet and Directed IPv4 packet wake up. It has the following format:

- Address 0x5840: IPV4 ADDR0
- Address 0x5848: IPV4 ADDR1
- Address 0x5850: IPV4 ADDR2
- Address 0x5858: IPV4 ADDR3

Table 37-135.IP4AT (0x5840 - 0x5858; RW)[0-3]: IPv4 Address Table Registers

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5840h at 8h **Offset End:** 5843h at 8h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5840h at 8h **Offset End:** 5843h at 8h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5840h at 8h **Offset End:** 5607h at 8h	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:** GbE0: Aux Gbe1/2: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IPV4ADDRx	IPV4 Address		Xh	RW

* Note: The first entry is loaded from the EEPROM if the IP Address Valid field of the EEPROM's Management Control word is 1 and the IP Address Type field is 0 (IPv4). Otherwise, the value of this register is undefined after reset

37.6.7.6 IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4

The IPv6 Address Table is used to store the IPv6 addresses for ARP/IPv6 Request packet and Directed IPv6 packet wake up and it has the following format

- Address 0x5880: IPV6 ADDR0, bytes 1 - 4 (Table 37-136)
- Address 0x5884: IPV6 ADDR0, bytes 5 - 8 (Table 37-137)
- Address 0x5888: IPV6 ADDR0, bytes 9 - 12 (Table 37-138)
- Address 0x588F: IPV6 ADDR0, bytes 13 - 16 (Table 37-139)

Table 37-136.IPV6_ADDR0BYTES_1_4 – IPv6 Address Table Register (0x5880), Bytes 1 - 4

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5880h **Offset End:** 5883h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5880h **Offset End:** 5883h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** **5880h** **Offset End:** 5883h	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IPV6ADDR0	IPV6 Address0, bytes 1 - 4		Xh	RW

37.6.7.7 IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8

The IPv6 Address Table is used to store the IPv6 addresses for ARP/IPv6 Request packet and Directed IPv6 packet wake up and it has the following format

- Address 0x5880: IPV6 ADDR0, bytes 1 - 4 (Table 37-136)
- Address 0x5884: IPV6 ADDR0, bytes 5 - 8 (Table 37-137)
- Address 0x5888: IPV6 ADDR0, bytes 9 - 12 (Table 37-138)
- Address 0x588F: IPV6 ADDR0, bytes 13 - 16 (Table 37-139)

Table 37-137.IPV6_ADDR0BYTES_5_8 – IPv6 Address Table Register, Bytes 5 - 8

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 05884h **Offset End:** 5887h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 05884h **Offset End:** 5887h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 05884h **Offset End:** 0588Fh	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IPV6ADDR1	IPV6 Address, bytes 5 - 8		X	RW

37.6.7.8 IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12

The IPv6 Address Table is used to store the IPv6 addresses for ARP/IPv6 Request packet and Directed IPv6 packet wake up and it has the following format

- Address 0x5880: IPV6 ADDR0, bytes 1 - 4 (Table 37-136)
- Address 0x5884: IPV6 ADDR0, bytes 5 - 8 (Table 37-137)
- Address 0x5888: IPV6 ADDR0, bytes 9 - 12 (Table 37-138)
- Address 0x588F: IPV6 ADDR0, bytes 13 - 16 (Table 37-139)

Table 37-138.IPV6_ADDR0BYTES_9_12 – IPv6 Address Table Register, Bytes 9 - 12

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 5888h **Offset End:** 588Bh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 5888h **Offset End:** 588Bh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 5888h **Offset End:** 588Bh	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IPV6ADDR2	IPV6 Address, bytes 9 - 12		X	RW

37.6.7.9 IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16

The IPv6 Address Table is used to store the IPv6 addresses for ARP/IPv6 Request packet and Directed IPv6 packet wake up and it has the following format

- Address 0x5880: IPV6 ADDR0, bytes 1 - 4 (Table 37-136)
- Address 0x5884: IPV6 ADDR0, bytes 5 - 8 (Table 37-137)
- Address 0x5888: IPV6 ADDR0, bytes 9 - 12 (Table 37-138)
- Address 0x588C: IPV6 ADDR0, bytes 13 - 16 (Table 37-139)

Table 37-139.IPV6_ADDR0BYTES_13_16 – IPv6 Address Table Register, Bytes 13 - 16

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 588Ch **Offset End:** 588Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 588Ch **Offset End:** 588Fh	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 588Ch **Offset End:** 588Fh	
Size: 32 bits	**Default:** XXXXXXXXh			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	IPV6DDR3	IPV6 Address, bytes 13 - 16		X	RW

Note: This table is loaded from the EEPROM if the IP Address Valid field of the EEPROM's Management Control word is 1 and the IP Address Type field is 1(IPv6). Otherwise, the value of this table is undefined after reset.

37.6.7.10 FFLT[0-3] – Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)

The Flexible Filter Length Table stores the minimum packet lengths required to pass each of the Flexible Filters. Any packets that are shorter than the programmed length won't pass that filter. Each Flexible Filter will consider a packet that doesn't have any mismatches up to that point to have passed the Flexible Filter when it reaches the required length. It will not check any bytes past that point:

- Address 0x5F00: FFLT_LEN0
- Address 0x5F04: Reserved
- Address 0x5F08: FFLT_LEN1
- Address 0x5F0C: Reserved
- Address 0x5F10 FFLT_LEN2
- Address 0x5F14: Reserved
- Address 0x5F18: FFLT_LEN3
- Address 0x5F1C: Reserved

Table 37-140.FFLT[0-3] - Flexible Filter Length Table Registers (0x5F00 - 0x5F18; RW)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 5F00h at 8h Offset End: 5F03h at 8h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 5F00h at 8h Offset End: 5F03h at 8h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 5F00h at 8h Offset End: 5F03h at 8h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 11	RSVD	Reserved		0h	RV
10 : 00	FFLT_LENx	Flexible Filter Length for FIlter x		0h	RW

Note: Before writing to the Flexible Filter Length Table the driver must first disable the flexible filters by writing 0's to the Flexible Filter Enable bits of the Wake Up Filter Control Register (WUFC.FLXn)

37.6.7.11 FFMT[0-127] – Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)

The Flexible Filter Mask and Table is used to store the four 1-bit masks for each of the first 128 data bytes in a packet, one for each Flexible Filter. If the mask bit is 1, the corresponding Flexible Filter will compare the incoming data byte at the index of the mask bit to the data byte stored in the Flexible Filter Value Table:

Table 37-141.Flexible Filter Mask Table

Address	Content	Address	Content
0x9000	Byte 0 Mask	0x9004	Reserved
0x9008	Byte 1 Mask	0x900C	Reserved
0x9010	Byte 2 Mask	0x9014	Reserved
...	Byte 3 - 126 Mask	...	Reserved
0x93F8	Byte 127Mask	0x93FC	Reserved

Table 37-142.FFMT[0-127] - Flexible Filter Mask Table Registers (0x9000 - 0x93F8; RW)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 9000h at 8h **Offset End:** 9003h at 8h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 9000h at 8h **Offset End:** 9003h at 8h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 9000h at 8h **Offset End:** 9003h at 8h	
Size: 32 bits	**Default:** 0000000Xh			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 04	RSVD	Reserved		0h	RV
03 : 00	Mask_x	Byte Mask for Byte xx		Xh	RW

Note: Before writing to the Flexible Filter Length Table the driver must first disable the flexible filters by writing 0's to the Flexible Filter Enable bits of the Wake Up Filter Control Register (WUFC.FLXn)

37.6.7.12 FFVT[0-127] – Flexible Filter Value Table Registers

The Flexible Filter Value and Table is used to store the one value for each byte location in a packet for each flexible filter. If the corresponding mask bit is 1, the Flexible Filter will compare the incoming data byte to the values stored in this table.

Table 37-143.Flexible Filter Mask Table

Address	**Content**	**Address**	**Content**
0x9800	Byte 0 Values	0x9804	Reserved
0x9808	Byte 1 Values	0x980C	Reserved
0x9810	Byte 2 Values	0x9814	Reserved
…	Byte 3 - 126 Values	…	Reserved
0x9BF8	Byte 127 Values	0x9BFC	Reserved

Table 37-144.FFVT[0-127]: Flexible Filter Value Table Registers

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 9800h at 8h Offset End: 9803h at 8h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 9800h at 8h Offset End: 9803h at 8h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 9800h at 8h Offset End: 9803h at 8h	
Size: 32 bits	Default: XXXXXXXXh			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 24	VAL3	Byte x Compare Value 3		X	RW
23 : 16	VAL2	Byte x Compare Value 2		X	RW
15 : 08	VAL1	Byte x Compare Value 1		X	RW
07 : 00	VAL0	Byte x Compare Value 0		X	RW

Note: Before writing to the Flexible Filter Length Table the driver must first disable the flexible filters by writing 0's to the Flexible Filter Enable bits of the Wake Up Filter Control Register (WUFC.FLXn)

37.6.8 Error Register Descriptions

37.6.8.1 INTBUS_ERR_STAT – Internal Bus Error Status Register

This register captures status information about errors that occur on the internal bus.

Table 37-145.INTBUS_ERR_STAT - Internal Bus Error Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 0510h Offset End: 0513h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 0510h Offset End: 0513h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 0510h Offset End: 0513h	
Size: 32 bits	Default: 00000000h			Power Well:	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 13	Rsvd	Reserved		0h	RV
12	INTBUS_ERR_H_DIS	0 - Internal Bus errors will halt further GbE transmit/ receive operation. 1 - Internal Bus errors will not halt further GbE operation.		0	RW

Table 37-145.INTBUS_ERR_STAT - Internal Bus Error Status Register (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0510h **Offset End:** 0513h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0510h **Offset End:** 0513h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0510h **Offset End:** 0513h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
11 : 06	Rsvd	Reserved		0h	RV
05 : 04	Type	Internal Bus **Error Type:** • 00 = Unsupported internal bus transaction targeted at GbE • 01 = Pull data error detected during a target write transaction • 10 = GbE received a Internal Bus Data Error response while mastering a DMA transaction • 11 = Master Pull data error occurred as a result of an internal memory error		0h	RO
03 :02	Rsvd	Reserved		0h	RV
01	MERR	Indicates whether one or more than one Internal Bus errors have occurred before INTBUS_ERR_STAT.CERR was cleared 0 = One Internal Bus Error 1 = More than one Internal Bus Error		0h	RWC
00	CERR	**Internal Bus Error:** Asserts when Internal Bus Error status and address registers are valid 0 = no error has been logged 1 = Internal Bus Error status and address registers have logged an error If error handling is enabled (INTBUS_ERR_H_DIS = 0) then this bit can only be cleared by a reset.		0h	RWC

37.6.8.2 MEM_TST – Memory Error Test Register

For software testability (in order to generate an ECC or parity error in hardware), this register can be used to generate a hardware ECC or parity error in the GbE memories. The selected ECC or parity error is continuously written in to the selected memory as long as the memory is selected, at any address currently in use.

The ‘Select’ field allows selection of the memory that the errored data should be written to. The ‘Mask’ field provides an XOR mask for the ECC or parity bits of that register. Although a 16 bit mask is provided, not all registers use all bits of the Mask field. Refer to the register description below.

The ECC or Parity error will not occur until the errored location has been read.

NOTE:

- Single bit errors will be corrected in all but types 010 and 011, with no error indication.
- Multiple bit errors will cause a Gbe error response for all other memory types.
- A multiple bit error is caused by forcing multiple mask bits [15:8] and/or multiple bits of [7:0] for Types 100, 101, 110. Forcing one bit in [15:8] and one bit in [7:0] will not create a multiple bit error (and will therefore not produce a Gbe Error response).

Table 37-146.MEM_TST - Memory Error Test Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0900h **Offset End:** 0903h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0900h **Offset End:** 0903h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0900h **Offset End:** 0903h	
Size: 32 bits	**Default:** 00000000h			**Power Well:**	GbE0: Aux Gbe1/2: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 19	Rsvd	Reserved		0h	RV
18 : 16	Select	Selects the memory where the error mask is applied: 000 : None - no errors injected 001 : Statistics Registers 010 : Multicast Filter Memory 011 : Special Packet Filter Memory 100 : TX Descriptor Buffer 101 : RX Descriptor Buffer 110 : Packet Buffer 111 : Flexible Filter Memory		0h	RW
15 : 00	Mask	ECC/Parity check bit XOR mask The Valid Mask bits are selected according to the Select field, as follows: 001 : 15:8 Reserved; 7:0 ECC Mask 010 : 15:4 Reserved; 3:0 Parity bit Mask 011 : 15:4 Reserved; 3:0 Parity bit Mask 100 : 15:0 ECC Mask 101 : 15:0 ECC Mask 110 : 15:0 ECC Mask 111 : 15:0 Reserved; 3:0 Parity bit Mask		0h	RW

37.6.8.3 MEM_STS – Memory Error Status Register

This register reports ECC or parity errors, for each of the memories with ECC or parity coverage. Errors will be reported in this register, regardless of whether they were induced by the MET logic or through actual hardware errors. Host write DMA transactions issued after these fatal memory errors are encountered will result in a data error asserted for every internal bus transaction. CSR target transactions, including transactions to the errored memory, will complete without error, however the memory errors will still be logged in the MES register.

If the MEM_ERRH_DIS bit is set (error handling is disabled), the individual memory errors can be cleared with a write to the appropriate memory error bit in this register, however the errored memory is likely to produce unexpected GbE functionality.

If the MEM_ERRH_DIS bit is not set (error handling is enabled) then the error bits will only be cleared by a soft reset.

Table 37-147.MEM_STS - Memory Error Status Register (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:0:0	**Offset Start:** 0904h **Offset End:** 0907h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:1:0	**Offset Start:** 0904h **Offset End:** 0907h	
View: PCI 3	**BAR:** CSRBAR		**Bus:Device:Function:** M:2:0	**Offset Start:** 0904h **Offset End:** 0907h	
Size: 32 bits	**Default:** 007F0000h			**Power Well:**	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 23	Rsvd	Reserved.		0h	RV
22 22	ERR_FLEX_DIS	Flex Filter Parity Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
21 21	ERR_STAT_DIS	Statistics Register ECC Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
20 20	ERR_PKBUF_DIS	Packet Buffer ECC Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
19 19	ERR_TXDS_DIS	Transmit Descriptor ECC Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
18 18	ERR_RXDS_DIS	Receive Descriptor ECC Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
17 17	ERR_SPF_DIS	Special Packets Filter Parity Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
16 16	ERR_MF_DIS	Multicast Filter Parity Error Disable 0: Error trapping enabled 1: Error trapping disabled		1h	RW
15 13	Rsvd	Reserved		0h	RV
12 12	MEM_ERRH_DIS	**Memory Error Handling Disable:** Indicates, for the following error types, whether GbE Tx/Rx operation will be halted: ERR_STAT ERR_PKBUF ERR_RXDS ERR_TXDS **0:** Memory Errors will halt further GbE Tx/Rx operation and a soft-reset is required to restore operation **1:** Memory Errors will be logged, but will not halt further GbE Tx/Rx operation		0h	RW
11 7	Rsvd	Reserved.		0h	RV

Table 37-147.MEM_STS - Memory Error Status Register (Sheet 2 of 2)

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:0:0	Offset Start: 0904h Offset End: 0907h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:1:0	Offset Start: 0904h Offset End: 0907h	
View: PCI 3	BAR: CSRBAR		Bus:Device:Function: M:2:0	Offset Start: 0904h Offset End: 0907h	
Size: 32 bits	Default: 007F0000h			Power Well:	GbE0: Aux Gbe1/2: Core

Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
6 6	ERR_FLEX	Flex filter Parity Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC.		0h	RO/RWC
5 5	ERR_STAT	Statistics Register ECC Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC		0h	RO/RWC
4 4	ERR_PKBUF	Packet Buffer ECC 2-bit Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC.		0h	RO/RWC
3 3	ERR_TXDS	Transmit Descriptor ECC 2-bit Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC.		0h	RO/RWC
2 2	ERR_RXDS	Receive Descriptor ECC 2-bit Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC.		0h	RO/RWC
1 1	ERR_SPF	Special Packets Filter Parity Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC		0h	RO/RWC
0 0	ERR_MF	Multicast Filter Parity Error 0: No error occurred 1: Error occurred When MEM_ERRH_DIS is clear then this bit is RO. When MEM_ERRH_DIS is set then this bit is RWC		0h	RO/RWC

37.7 Power Management

Note: Power Management may be disabled via bits in the Initialization Control Word which is loaded from the EEPROM during power-up reset. Even when disabled, the Power Management register set is still present.

The following Power Management related features are supported:

- Power states of D0 & D3hot
- Support of optional D3cold for GbE0
- Power(D3) < Power(D0)
- Wake-up

37.7.1 Assumptions

The following assumptions apply:

- Prior to transition from D0 to the D3 state, the OS will ensure the device driver has been disabled and all pending bus transactions are complete or terminated cleanly.
- Per the PCI power Management Specification (revision 1.1, section 5.4), software "will need to perform a full re-initialization of the function including its PCI Configuration Space." **On a transition from D3 to D0u state, all of the PCI Configuration space is reset.**
- The driver will set up the wake up filters prior to the system transitioning the GbE to D3 state.
- No wakeup capability, except APM Wakeup if enabled in the EEPROM, is required after the system puts the MAC in D3 state and then returns the MAC to D0.
- If the APMPME bit in the Wake Up Control Register (WUC.APMPME) bit is 1, it is permissible to assert GBE_PME_WAKE even when PME_En is 0.
- No wakeup capability, except APM Wakeup if enabled in the EEPROM, is required after the system asserts, then deasserts RESET_N.
- The deassertion (rising) edge of RESET_N will put the controller in D0u state.
- The system will never deactivate the internal bus clock without asserting RESET_N.
- Any time PWR_OK is asserted all power supplies are stable and RESET_N is stable.

37.7.2 D3cold support

If the D3Cold Wake Up Capability Advertisement Enable bit of the Device Control Register (CTRL.ADVD3WUC) is set to '1', the D3Cold capability may be advertised (if 0, the capability will not be advertised). When 1, the EP80579 will then use the AUX_PWR_PRESENT input as an indication of whether auxiliary power is available to the GbE controller, and if AUX_PWR_PRESENT=1 will advertise D3cold Wake Up support in the PCI Capabilities list for that GbE controller.

If D3cold is supported, the PME_En and PME_Status bits of the Power Management Control/Status Register (PMCSR), as well as shadow bits in the Wake Up Control Register (WUC), are not reset by UNIT_RESET. However, if D3cold Wake Up is not supported they will always be reset on the deassertion (falling edge) of UNIT_RESET.

Note that AUX_PWR_PRESENT is level sensitive and is sampled at assertion of AUX_PWR_GOOD from the EEDI pin. EEDI must be pulled high to indicate that AUX_PWR_PRESENT is a '1'. EEDI must be pulled low to indicate that AUX_PWR_PRESENT is a "0'. EEDI MUST be either pulled up or down. AUX_PWR_PRESENT may not be left indeterminate.

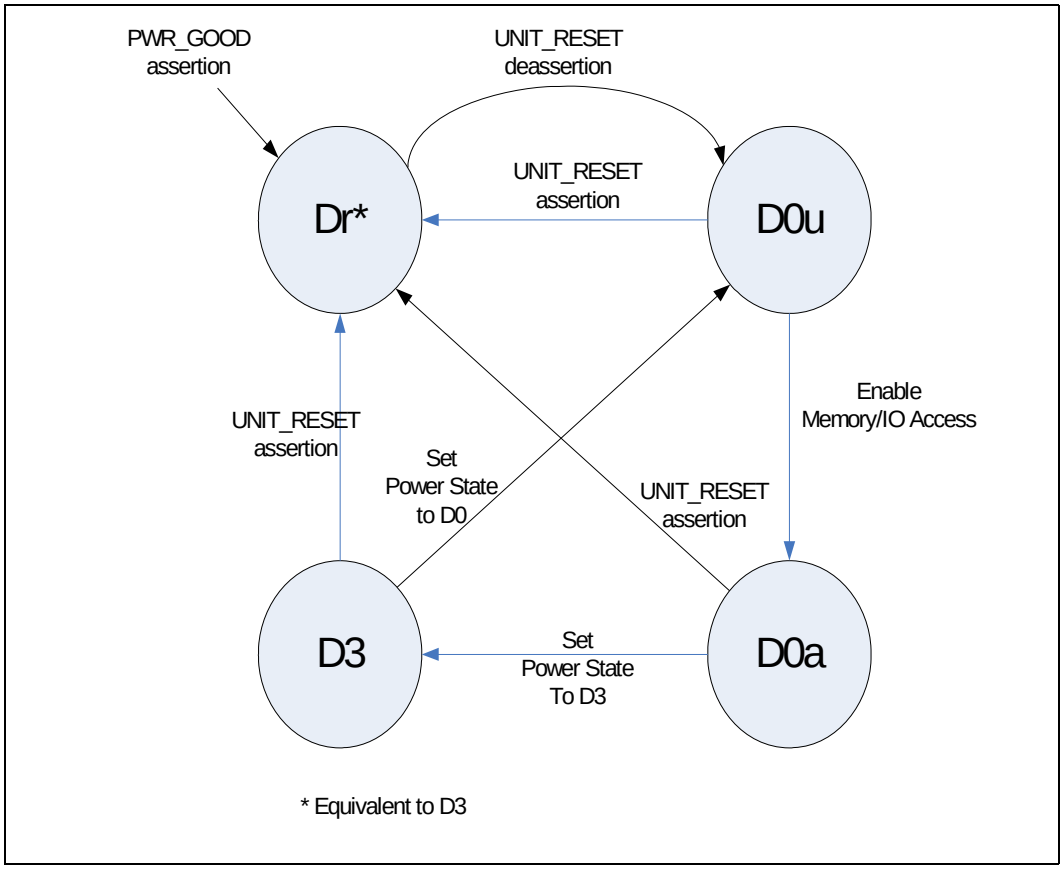
Besides the reset change and capability advertisement, the EP80579 D3cold behavior is identical to D3hot behavior (i.e. operates the same regardless of whether power is supplied from main vs. auxiliary power supplies).

37.7.3 Power States

The GbE Controller supports D0 and D3 power states defined in the PCI Power Management Specification. In addition, D0 is divided into two sub-states: D0u, and D0a. In addition, it supports a Dr state that is entered when UNIT_RESET is asserted.

Figure 37-50 shows the power states and what causes a transition between them.

Figure 37-50.Power State Transitions



37.7.3.1 Dr

On initial boot-up, once PWR_GOOD is asserted the GbE controller will enter the Dr state and read the EEPROM. If the APM Enable bit in the EEPROM's Initialization Control Word 2 is set then APM Wake Up will be enabled.

Internally, the controller treats the Dr reset state equivalently to D3. Any WakeUp filter settings that were enabled before entering this reset state will be maintained.

The deassertion (falling edge) of UNIT_RESET will cause a transition to D0u.

37.7.3.2 D0u State

The D0u state is a low-power state used after UNIT_RESET is deasserted, or when coming out of D3, but before the GbE Controller is initialized.

When entering D0u the GbE Controller will disable Wake Ups, assert reset to the PHY for between 24µs and 1ms, and re-read the EEPROM. If the APM Mode bit in the EEPROM's Initialization Control Word 2 is set then APM Wake Up will be enabled.

On a transition from D3 to D0u state, all of the PCI Configuration space is reset. Per the PCI Power Management Specification (revision 1.1, section 5.4), software “will need to perform a full reinitialization of the function including its PCI Configuration Space.”

Internally, D0u is treated like D3 and some internal clocks and registers are shut down. As in the Dr state, the PHY power state depends on whether PHY power-management and WakeUp are enabled.

The D0u state is exited when the system enables memory space access to the controller by writing a 1 to the Memory Access Enable bit of the PCI Command Register.

37.7.3.3 D0a

Once memory space is enabled all internal clocks are activated and the controller enters an active state. It can transmit and receive packets if properly configured by the driver. Any APM Wakeup previously active will remain active. The driver can deactivate APM Wakeup by writing to the Wake Up Control Register (WUC), or activate other Wake Up Filters by writing to the Wake Up Filter Control Register (WUFC).

37.7.3.4 D3

Prior to transition from D0 to the D3 state, the driver must ensure that controller transmit and receive operation has been disabled and all pending bus transactions are complete or terminated cleanly. If wake up capability is needed the driver should set up the appropriate wake up registers and the system should write a 1 to the PME_En bit of the Power Management Control / Status Register (PMCSR) prior to the transition to D3.

When the system writes a '11 to the PowerState field of the Power Management Control/Status Register (PMCSR) the GbE controller will transition to D3. Any WakeUp filter settings that were enabled before entering this reset state will be maintained. Upon transitioning to D3 the controller will clear the Memory Access Enable bit of the PCI Command Register, which will disable memory access decode.

For power savings, the GbE controller shuts down some internal clocks and registers in D3 state.

To transition back to D0u, the system writes a 00 to the Power State field of the Power Management Control/Status Register (PMCSR).

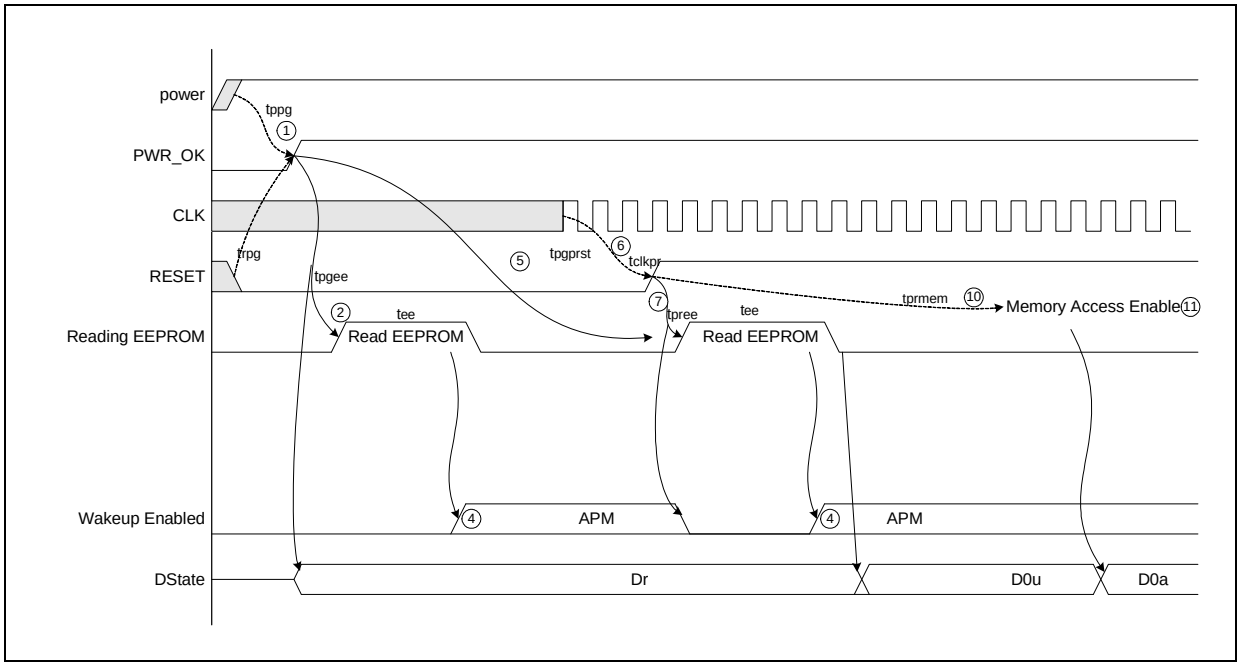
37.7.4 Timing of Power-State Transitions

The following sections give detailed timing for the state transitions. In the diagrams the dotted connecting lines represent the MACs’ requirements, while the solid connecting lines represent the MACs’ guarantees.

The timing diagrams are not to scale. The clocks edges are shown to indicate running clocks only are not used to indicate the actual number of cycles for any operation.

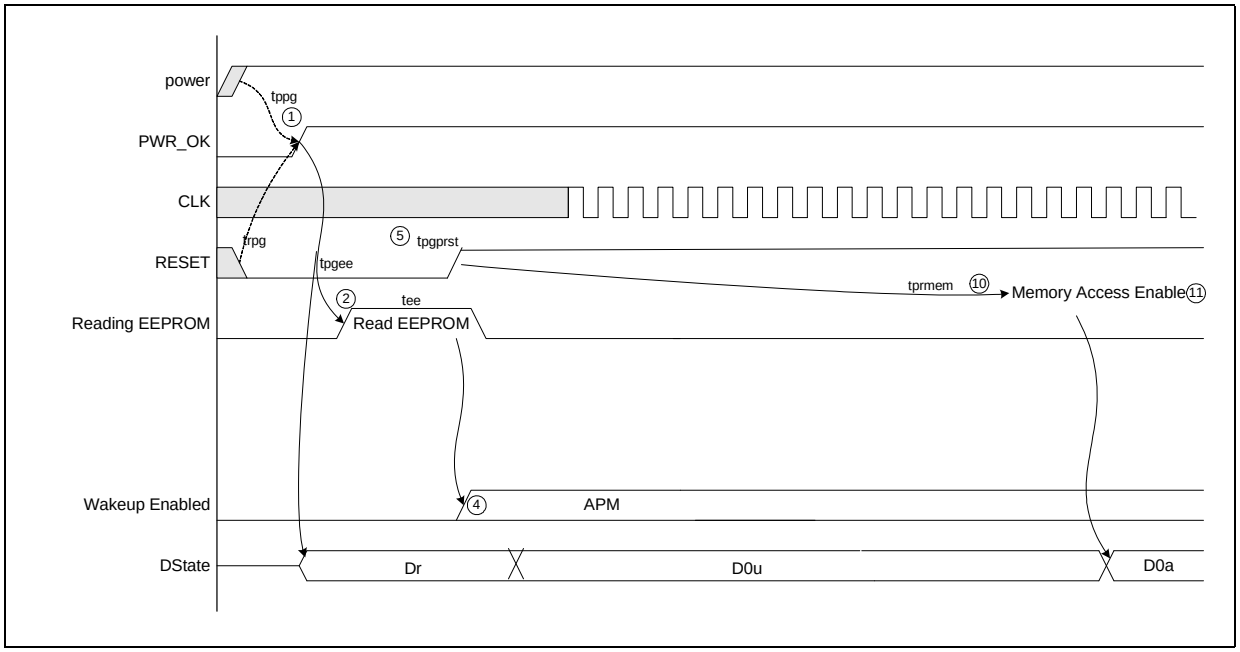
37.7.4.1 Power up (off to Dr to D0u to D0a)

Figure 37-51.Reset Deasserted after 1st EEPROM Read Completes



Note	
1	PWR_OK must not be asserted until all power supplies are good
2	An EEPROM read is started on the rising edge of PWR_OK and RESET.
4	APM Wakeup may be enabled based on what is read from the EEPROM.
5	The system can delay an arbitrary time before deasserting RESET.
7	The deassertion edge of RESET will cause the EEPROM to be re-read and Wake Up disabled.
10	The system can delay an arbitrary time before enabling Memory Access.
11	Writing a 1 to the Memory Access Enable bit in the PCI Command Register will transition the MAC from D0u to D0 state

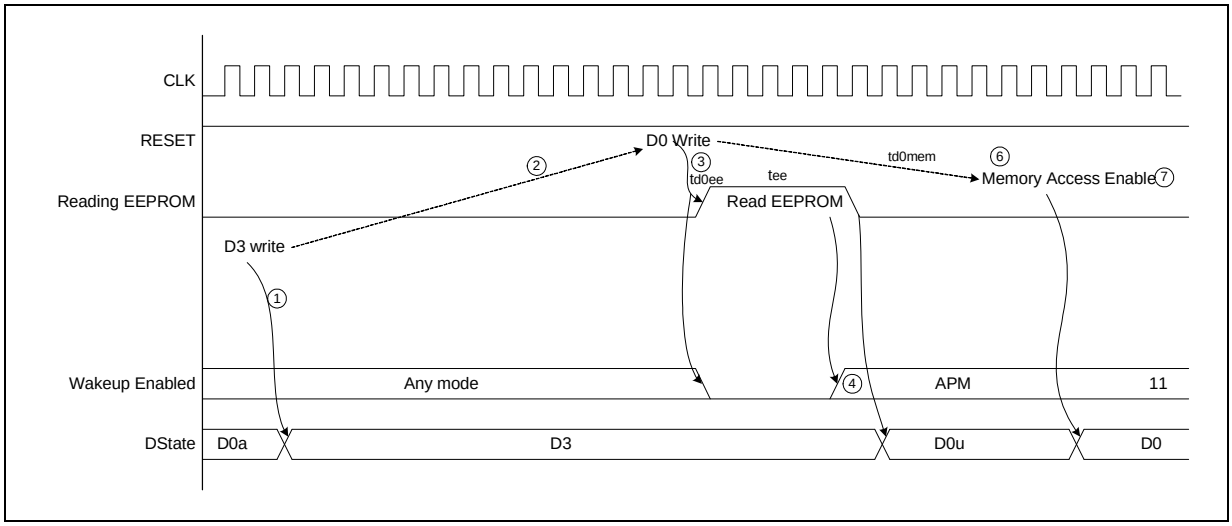
Figure 37-52.Reset Deasserted after before EEPROM Read Completes



Note	
1	PWR_OK must not be asserted until all power supplies are good
2	An EEPROM read is started on the rising edge of PWR_OK and RESET.
4	APM Wakeups may be enabled based on what is read from the EEPROM.
5	The system can delay an arbitrary time before deasserting RESET.
10	The system can delay an arbitrary time before enabling Memory Access.
11	Writing a 1 to the Memory Access Enable bit in the PCI Command Register will transition the MAC from D0u to D0 state

37.7.4.2 Transition from D0a to D3 and Back without Reset

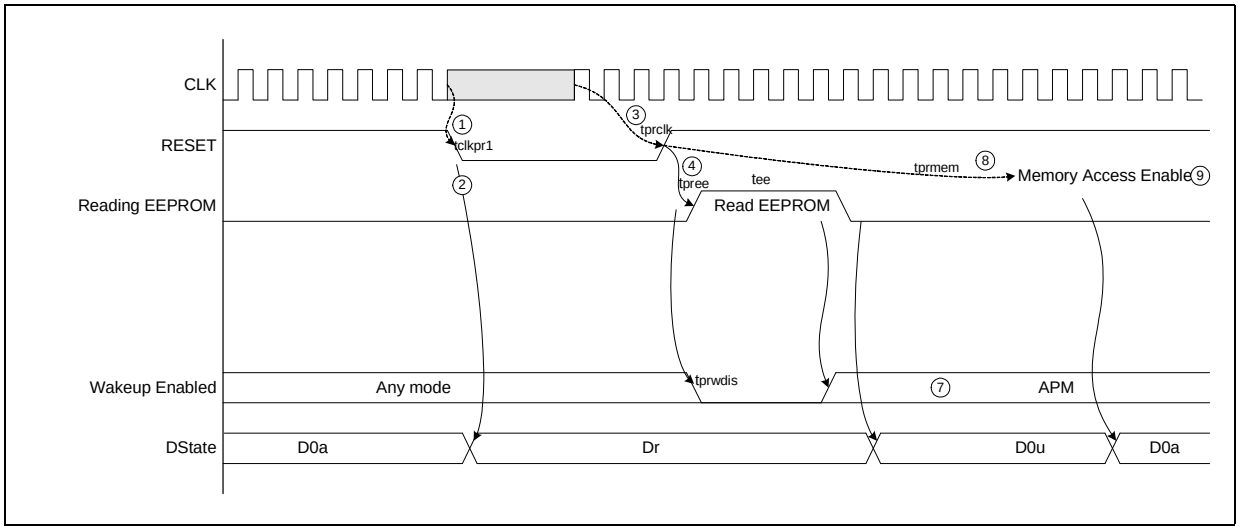
Figure 37-53.Transition from D0a to D3 and Back without Reset



Note	
1	Writing a 11 to the Power State field of the Power Management Control/Status Register (PMCSR) will transition the power state to D3.
2	The system can delay an arbitrary amount of time between setting D3 mode and asserting RESET.
4	Upon assertion of RESET the MAC will go to “Dr” state.
6	The deassertion edge of RESET will case the EEPROM to be re-read and Wake Up disabled.
10	The system can delay an arbitrary time before enabling memory access.
11	Writing a 1 to the Memory Access Enable bit in the PCI Command Register will transition the MAC from D0u to D0 state

37.7.4.3 Transition from D0a to D3 and Back with Reset

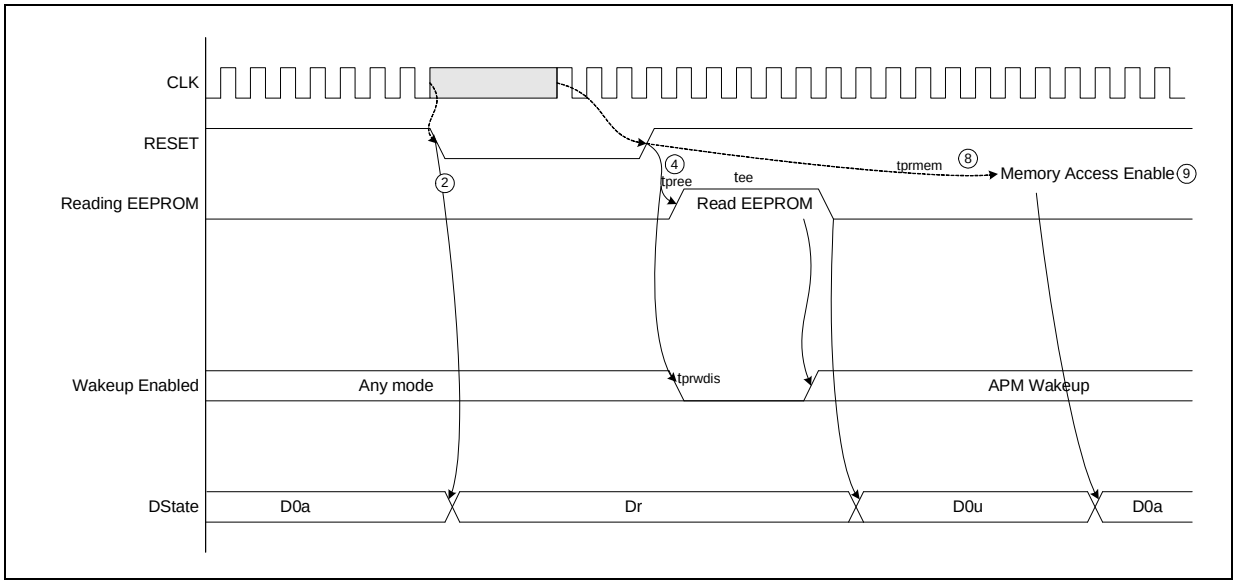
Figure 37-54.Transition from D0a to D3 and Back with Reset



Note	
1	Writing a 11 to the Power State field of the Power Management Control/Status Register (PMCSR) will transition the MAC to D3.
2	The system can delay an arbitrary amount of time between setting D3 mode and asserting RESET.
4	Upon assertion of RESET the MAC will go to “Dr” state.
6	The deassertion edge of RESET will case the EEPROM to be re-read and Wake Up disabled.
10	The system can delay an arbitrary time before enabling memory access.
11	Writing a 1 to the Memory Access Enable bit in the PCI Command Register will transition the MAC from D0u to D0 state.

37.7.4.4 Reset without Transition to D3

Figure 37-55.Reset without Transition to D3



Note	
2	Upon assertion of RESET the MAC will go to “Dr” state.
4	The deassertion edge of RESET will case the EEPROM to be re-read, and Wake Up disabled.
8	The system can delay an arbitrary time before enabling memory access.
9	Writing a 1 to the Memory Access Enable bit in the PCI Command Register will transition the MAC from D0u to D0 state.

37.7.4.5 Timing Requirements

The MAC requires the following start-up or power state transition related timing

Table 37-148.MAC Timing

Parameter	Description	Min	Max.	Notes
tppg	Power to PWR_OK	0	-	
trpg	RESET stable to PWR_OK	0	-	
tpgrst	PWR_OK assertion to RESET deassertion.	0	-	
tprmem	RESET deassertion to Memory Access Enable	10ms	-	The Power Management specification permits access in 10ms after exiting D3 cold. PCI 2.2 gives 225 cycles.
tclkpr1	CLK stopped to RESET deassertion.	0ns		The reset must be asserted before the rising edge of the last cycle before the clock is stopped.

37.7.4.6 Timing Guarantees

The GbE guarantees the following start-up or power state transition related timing parameters.

Table 37-149.GbE Timing Guarantees

Parameter	Description	Min	Max.	Notes
tpgee	PWR_OK assertion to start of EEPROM read.	0	1µs	
tpree	RESET deassertion to start of EEPROM read	0	10µs	
tee	EEPROM read duration	24µs	10ms	The MAC must attempt to read the EEPROM to determine if an EEPROM is present, so this applies even if no EEPROM is connected.
td3ps	D3 write to power reduction.	0	500ns	

37.7.5 Power Management Extended Capabilities Registers

Power Management registers are part of the capabilities linked list pointed to by the Capabilities Pointer (Cap_Ptr) in the PCI configuration space.

All fields are reset by PWR_GOOD. All of the fields except PME_En and PME_Status are reset by the deassertion (rising edge) of UNIT_RESET. If AUX_PWR_PRESENT=0, the PME_En and PME_Status fields also reset by the deassertion (rising edge) of UNIT_RESET.

Refer to Section 35.6.1.17, “Offset DCh: PCID – Power Management Capability ID Register” on page 1251, Section 35.6.1.18, “Offset DDh: PCP – Power Management Next Capability Pointer Register” on page 1251, Section 35.6.1.19, “Offset DEh: PMCAP – Power Management Capability Register” on page 1252, and Section 35.6.1.20, “Offset E0h: PMCS – Power Management Control and Status Register” on page 1253 for register details.

§ §

38.0 Global Configuration Unit

38.1 Overview

The Global Configuration Unit (GCU) implements the configuration space registers for system-level, EP80579 features. This unit also implements an MDIO interface to support the Ethernet interfaces.

38.2 Feature List

The following is a list of features/register sets implemented in the Global Configuration Unit:

- MDIO interface
  - Command and Status registers
  - MDIO Drive Strength register
  - MDC Drive Strength register
- RCOMP registers
  - Local Expansion bus (LEB)
  - GbE (one set for all three interfaces)
- Drive Strength registers
  - SSP Signal Group Drive Strength register
  - TDM outputs
  - CAN (one for both)

38.3 Usage Model

All configuration writes and reads are word accesses. Addresses to configuration reads and writes are word aligned.

38.3.1 RCOMP

The RCOMP circuitry dynamically compensates the IO output drivers for variations in operating conditions due to process, temperature, voltage and PCB layout. These variations are measured through a resistive mechanism in two special IO pads. The resistive mechanism on those IO pads reference external resistors the user provides to match output driver strength to. Thus the output driver impedance can be tuned specifically to the application PCB characteristics for nominal signal transfer into the transmission lines formed by the PCB traces. The RCOMP design is nominally set to operate at 50ohms, but the user is free to set the impedance in the range 45ohms to 55ohms.

Two RCOMP pins are provided to establish the output driver impedance, one to control the drive high strength, and one to control the drive low strength. These RCOMP outputs drive into the external resistors provided by the customer. The drivers form a resistor divider and the voltages developed in these dividers are compared to a reference voltage. The RCOMP state machine independently adjusts the strength of the drivers making them stronger or weaker until the comparator signals that the voltage is greater than or less than the reference. The state machine will continue making adjustments causing the comparators to oscillate between two strength settings just above and just below the comparator trip point. Logic in the RCOMP state machine recognizes when this "dithering" between the two values has begun, and holds the strength output for the outputs fixed at one of the two settings. Note this algorithm is independently and concurrently applied to the drive high strength and to the drive low strength.

38.3.1.1 GbE

The GbE RCOMP controller starts operation when the GBE_AUX_PWR_GOOD input is asserted. This condition indicates the power supply's are stable and it also indicates that the GBE_REFCLK input is being driven with a 125 MHz clock. Refer to Section 37.0, “Gigabit Ethernet Controller” on page 1341 for details on the GbE operation. The GBE_REFCLK input is divided by either 4 or 16 and then used to drive the RCOMP state machine. Also present are software accessible registers that provide options to monitor/overwrite internal bias and comparator output for SV needs. See Section 38.4.1.5, “Offset 0x00000024h: GCU_GBE_RC_CTRL - GCU GbE RCOMP Control Register” on page 1564 and Section 38.4.1.6, “Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register” on page 1564.

38.3.1.2 LEB

The LEB RCOMP controller starts operation when the LEB comes out of reset. The RCOMP logic is clocked by the externally provided expansion bus clock. Refer to Section 42.0, “Local Expansion Bus Controller” on page 1671 for details on the LEB operation. Also present are software accessible registers that provide options to monitor/overwrite internal bias and comparator output for SV needs. See Section 38.4.1.7, “Offset 0x00000050h: GCU_LEB_RC_STAT - GCU Local Expansion Bus RCOMP Status Register” on page 1565 and Section 38.4.1.8, “Offset 0x00000054h: GCU_LEB_RC_CTRL - GCU Local Expansion Bus RCOMP Control Register” on page 1566

38.4 Register Summary

Writes to unused address space have no affect. Reads from unused address space may return indeterminate data. Neither reads nor writes cause detrimental effects on device operation unless specifically documented. Reserved bits within registers must be written with their reset value unless otherwise stated.

For more information on the conventions the following register summaries adopt, see Section 7.1, "Overview of Register Descriptions and Summaries" on page 183.

Note: Any programming order dependencies on configuration write values must be handled by the OS.

The Global Configuration Unit registers materialize in the PCI space.

Table 38-1. Bus M, Device 3, Function 0: Summary of GCU Registers Mapped Through CSRBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00000010h	00000013h	"Offset 0x00000010h: MDIO_STATUS - MDIO Status Register" on page 1562	00000000h
00000014h	00000017h	"Offset 0x00000014h: MDIO_COMMAND - MDIO Command Register" on page 1562	00000000h
00000018h	0000001Bh	"Offset 0x00000018h: MDIO_DRIVE - MDIO Drive Register" on page 1563	03030107h
00000020h	00000023h	"Offset 0x00000020h: MDC_DRIVE - MDC Drive Register" on page 1563	0303030Fh
00000024h	00000027h	"Offset 0x00000024h: GCU_GBE_RC_CTRL - GCU GbE RCOMP Control Register" on page 1564	0031F31Fh
00000044h	00000047h	"Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register" on page 1564	00000000h
00000050h	00000053h	"Offset 0x00000050h: GCU_LEB_RC_STAT - GCU Local Expansion Bus RCOMP Status Register" on page 1565	63000300h
00000054h	00000057h	"Offset 0x00000054h: GCU_LEB_RC_CTRL - GCU Local Expansion Bus RCOMP Control Register" on page 1566	000030F301h
00000060h	00000063h	"Offset 0x00000060h: SSP_DRIVE - SSP Drive Register" on page 1566	02000200h
00000064h	00000067h	"Offset 0x00000064h: TDM_DRIVE_3 - TDM Drive Register for TDM ports 3" on page 1567	02000200h
00000068h	0000006Bh	"Offset 0x00000068h: TDM_DRIVE_12 - TDM Drive Register for TDM ports 1 & 2" on page 1567	02000200h
00000028h	0000002Bh	"Offset 0x00000028h: CAN_DRIVE - CAN Drive Register" on page 1568	02000200h

Table 38-2. Register-Table Legend

Attribute	Legend
RV	Reserved
RO	Read Only
RW	Read/Write
WO	Write Only
RS	Set automatically when read
RC	Cleared automatically when read
WC	Write to clear. See individual bit description for more details.

38.4.1 Detailed Register Descriptions

38.4.1.1 Offset 0x00000010h: MDIO_STATUS - MDIO Status Register

Table 38-3. Offset 0x00000010h: MDIO_STATUS - MDIO Status Register

Description:	Status register for communicating with MDIO devices.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000010h **Offset End:** 00000013h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	MDIO_STATUS	0 = Successfully read 1 = Read Error		0b	RO
30 :16	RSVD	Reserved		0000h	RO
15 :00	MDIO_READ_DATA	Read Data		0000h	RO

38.4.1.2 Offset 0x00000014h: MDIO_COMMAND - MDIO Command Register

Table 38-4. Offset 0x00000014h: MDIO_COMMAND - MDIO Command Register

Description:	Command register for communicating with MDIO devices.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000014h **Offset End:** 00000017h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	MDIO_Go	Application logic sets this to “1” to start the MDIO access. This bit remains “1” during the access. When the access is finished, this bit is reset to “0”.		0b	RW
30 :27	RSVD	Reserved for future use.		0h	RV
26	MDIO_Oper	1 = MDIO Write Access 0 = MDIO Read Access Note: Once the transaction completes (as indicated by MDIO_Go returning “0”), this bit is also reset to ‘0’.		0b	RW
25 :21	MDIO_PHY_ADDR	Physical address of the PHY to be accessed.		0h	RW
20 :16	MDIO_PHY_REG	Register number of the PHY register to be accessed. Note: Once the transaction completes (as indicated by MDIO_Go returning “0”), this field is also reset to ‘0’.		00h	RW
15 :00	MDIO_WRITE_DATA	Write data on MDIO write accesses. Note: Once the transaction completes (as indicated by MDIO_Go returning “0”), this field is also reset to ‘0’.		0000h	RW

38.4.1.3 Offset 0x00000018h: MDIO_DRIVE - MDIO Drive Register

Table 38-5. Offset 0x00000018h: MDIO_DRIVE - MDIO Drive Register

Description:	Drive Control of MDIO output bits.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000018h **Offset End:** 0000001Bh	
Size: 32 bit	**Default:** 03030107h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	RSVD	Reserved		00h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		03h	RW
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		03h	RW
15 :12	RSVD	Reserved		00h	RV
11 :08	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		01h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		07h	RW

38.4.1.4 Offset 0x00000020h: MDC_DRIVE - MDC Drive Register

Table 38-6. Offset 0x00000020h: MDC_DRIVE - MDC Drive Register

Description:	Drive Control of MDc outputs.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000020h **Offset End:** 00000023h	
Size: 32 bit	**Default:** 0303030Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	RSVD	Reserved		00h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		03h	RW
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		03h	RW
15 :12	RSVD	Reserved		00h	RV
11 :08	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		03h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		0Fh	RW

38.4.1.5 Offset 0x00000024h: GCU_GBE_RC_CTRL - GCU GbE RCOMP Control Register

Table 38-7. Offset 0x00000024h: GCU_GBE_RC_CTRL - GCU GbE RCOMP Control Register

Description:	RCOMP Control for GbE Bus outputs.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000024h **Offset End:** 00000027h	
Size: 32 bit	**Default:** 0031F31Fh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	EN	On-die termination disabled. 0 - Enabled 1 - Not enabled		0h	RW
30	OVR_WR	Overwrite RCOMP engine strength outputs with DN_TST and UP_TST and apply to output pads		0h	RW
29	RSVD	Reserved		0h	RV
28	RSVD	Reserved		0h	RV
27	RSVD	Reserved		0h	RV
26	MNTR	Monitor. Disable the update lowpass filter. Allow new strength values to be output to the rcomp pads as soon as they are calculated.		0h	RW
25	STOP	Stop sending new strength values to the rcomp pads. MNTR has precedence over STOP		0h	RW
24	DIV_BPS	When set, rcomp engine runs at GBE_REFCLK / 4. When clear, rcomp engine runs at GBE_REFCLK/16		0h	RW
23 :20	DN_TST_11_8	Pull down strength bits [11:8]. Each bit enables 5 unit pull-down drivers in the output pad.		3h	RW
19 :12	DN_TST_7_0	Pull down strength bits [7:0]. Each bit enables 1 unit pull-down driver in the output pad.		1Fh	RW
11 :08	UP_TST_11_8	Pull up strength bits [11:8]. Each bit enables 5 unit pull-up drivers in the output pad.		3h	RW
7 :00	UP_TST_7_0	Pull up strength bits [7:0]. Each bit enables 1 unit pull-up driver in the output pad.		1Fh	RW

38.4.1.6 Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register

Table 38-8. Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register

Description:	RCOMP status of Local Expansion Bus outputs.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:3:0		**Offset Start:** 00000044h **Offset End:** 00000047h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Ready	Ready from the RCOMP controller		0h	RO
30	PMOS_COMP	PMOS comparator output (input to controller)		0h	RO
29	NMOS_COMP	NMOS comparator output (input to controller)		0h	RO

Table 38-8. Offset 0x00000044h: GCU_GBE_RC_STAT - GCU GbE RCOMP Status Register

Description:	RCOMP status of Local Expansion Bus outputs.				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000044h **Offset End:** 00000047h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
28	RSVD	Reserved		0h	RO
27 :16	NMOS_CTRL	NMOS strength control output to pads		0h	RO
15 :12	RSVD	Reserved		0h	RO
11 :00	PMOS_CTRL	PMOS strength control output to pads		0h	RO

38.4.1.7 Offset 0x00000050h: GCU_LEB_RC_STAT - GCU Local Expansion Bus RCOMP Status Register

Table 38-9. Offset 0x00000050h: GCU_LEB_RC_STAT - GCU Local Expansion Bus RCOMP Status Register

Description:	RCOMP status of Local Expansion Bus outputs.				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000050h **Offset End:** 00000053h	
Size: 32 bit	**Default:** 63000300h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Ready	Ready from the RCOMP controller		0h	RO
30	PMOS_COMP	PMOS comparator output (input to controller)		1h	RO
29	NMOS_COMP	NMOS comparator output (input to controller)		1h	RO
28	RSVD	Reserved		0h	RO
27 :16	NMOS_CTRL	NMOS strength control output to pads		300h	RO
15 :12	RSVD	Reserved		0h	RO
11 :00	PMOS_CTRL	PMOS strength control output to pads		300h	RO

38.4.1.8 Offset 0x00000054h: GCU_LEB_RC_CTRL - GCU Local Expansion Bus RCOMP Control Register

Table 38-10. Offset 0x00000054h: GCU_LEB_RC_CTRL - GCU Local Expansion Bus RCOMP Control Register

Description:	RCOMP Control for GbE Bus outputs.				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000054h **Offset End:** 00000057h	
Size: 32 bit	**Default:** 000030F301h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	SW_RST	Software reset Write '1' to reset. Write '0' to enable operation		0h	RW
30	OVR_WR	Overwrite output bus values from special register		0h	RW
29	CMP	For overwrite comparator output		0h	RW
28	CMPO_WR	For overwrite comparator output (select)		0h	RW
27	CMPO_RD	For test comparator output		0h	RW
26	MNTR	Bypass the update logic system		0h	RW
25	STOP	"Freeze" output bus, but circuit operation continue		0h	RW
24	DIV_BPS	Bypass internal clock divider		0h	RW
23 :20	DN_TST_11_8	Pull down strength bits [11:8]. Each bit enables 5 unit pull-down drivers in the output pad.		3h	RW
19 12	DN_TST_7_0	Pull down strength bits [7:0]. Each bit enables 1 unit pull-down driver in the output pad.		0Fh	RW
11 8	UP_TST_11_8	Pull up strength bits [11:8]. Each bit enables 5 unit pull-up drivers in the output pad.		3h	RW
7 0	UP_TST_7_0	Pull up strength bits [7:0]. Each bit enables 1 unit pull-up driver in the output pad.		01h	RW

38.4.1.9 Offset 0x00000060h: SSP_DRIVE - SSP Drive Register

Table 38-11. Offset 0x00000060h: SSP_DRIVE - SSP Drive Register

Description:	Drive Control of SSP outputs.				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000060h **Offset End:** 00000063h	
Size: 32 bit	**Default:** 02000200h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :30	RSVD	Reserved		0h	RV
29 :28	RSVD	Reserved		0h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		02h	RW
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		0h	RW

Table 38-11. Offset 0x00000060h: SSP_DRIVE - SSP Drive Register

Description:	Drive Control of SSP outputs.				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000060h **Offset End:** 00000063h	
Size: 32 bit	**Default:** 02000200h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
15 :12	RSVD	Reserved		0h	RV
11 :8	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		02h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		0h	RW

38.4.1.10 Offset 0x00000064h: TDM_DRIVE_3 - TDM Drive Register for TDM Port 3

Table 38-12. Offset 0x00000064h: TDM_DRIVE_3 - TDM Drive Register for TDM ports 3

Description:	Drive Control of TDM port 3outputs				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000064h **Offset End:** 00000067h	
Size: 32 bit	**Default:** 02000200h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	RSVD	Reserved		0h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		02h	RW
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		0h	RW
15 :12	RSVD	Reserved		0h	RV
11 :8	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		02h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		0h	RW

38.4.1.11 Offset 0x00000068h: TDM_DRIVE_12 - TDM Drive Register for TDM Ports 1 & 2

Table 38-13. Offset 0x00000068h: TDM_DRIVE_12 - TDM Drive Register for TDM ports 1 & 2

Description:	Drive Control of TDM port 1 & 2outputs				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:3:0	**Offset Start:** 00000068h **Offset End:** 0000006Bh	
Size: 32 bit	**Default:** 02000200h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	RSVD	Reserved		0h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		02h	RW

Table 38-13. Offset 0x00000068h: TDM_DRIVE_12 - TDM Drive Register for TDM ports 1 & 2

Description:	Drive Control of TDM port 1 & 2outputs				
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:3:0	Offset Start: 00000068h Offset End: 0000006Bh	
Size: 32 bit	Default: 02000200h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		0h	RW
15 :12	RSVD	Reserved		0h	RV
11 :08	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		02h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		0h	RW

38.4.1.12 Offset 0x00000028h: CAN_DRIVE - CAN Drive Register

Table 38-14. Offset 0x00000028h: CAN_DRIVE - CAN Drive Register

Description:	Drive Control of both CAN controller outputs.				
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:3:0	Offset Start: 00000028h Offset End: 0000002Bh	
Size: 32 bit	Default: 02000200h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :28	RSVD	Reserved		0h	RV
27 :24	mdio_nchan_5x	Drive strength for N-channel at 5x ratio		02h	RW
23 :16	mdio_nchan_1x	Drive strength for N-channel at 1x ratio		0h	RW
15 :12	RSVD	Reserved		0h	RV
11 :8	mdio_pchan_5x	Drive strength for P-channel at 5x ratio		02h	RW
07 :00	mdio_pchan_1x	Drive strength for P-channel at 1x ratio		0h	RW

§ §

39.0 Controller Area Network - CAN

39.1 Overview

Controller Area Network (CAN) is a serial bus system used in a broad range of embedded and automation control systems. The CAN usually links two or more microcontroller-based physical devices (nodes).

CAN protocol is based on a broadcast communication mechanism. This broadcast communication is achieved by using a message-oriented transmission protocol. In this protocol, node-addresses are not defined, only messages are defined. These messages are identified by a message identifier. The message identifier has to be unique within the network and it defines not only the content, but also the priority of the message.

A high degree of system and configuration flexibility is achieved as a result of the content-oriented addressing scheme.This content-oriented addressing scheme allows for a high degree of system and configuration flexibility. It is easy to add stations to an existing CAN network without making any hardware or software modifications to the existing stations, as long as the new stations are purely receivers. This allows for modularization of the components and also permits multiple reception and the synchronization of distributed processes: data needed as information by several stations can be transmitted via the network in such a way that it is unnecessary for each station to know who the producer of the data is. This allows easy servicing and upgrading of networks, as data transmission is not based on the availability of specific types of stations.

39.2 Feature List

Key features are listed below:

- Conforms to ISO 11898-1(high-speed CAN).
- Full CAN 2.0B compliance.
  - — Supports CAN 2.0A.
- Supports 11-bit identifier and 29-bit identifier.
- Bit rates up to 1Mbps.
- 16 receive buffers:
  - — Each buffer has its own message filter.
  - — Message filter covers: ID, IDE, RTR, Data Byte (1 and 2).
  - — Message buffers can be linked together to build a larger message array.
  - — Automatic RTR response handler.
- Transmit path:
  - — 8 transmit message holding registers with programmable priority arbitration.
  - — Message abort command.
- Test and debugging support:

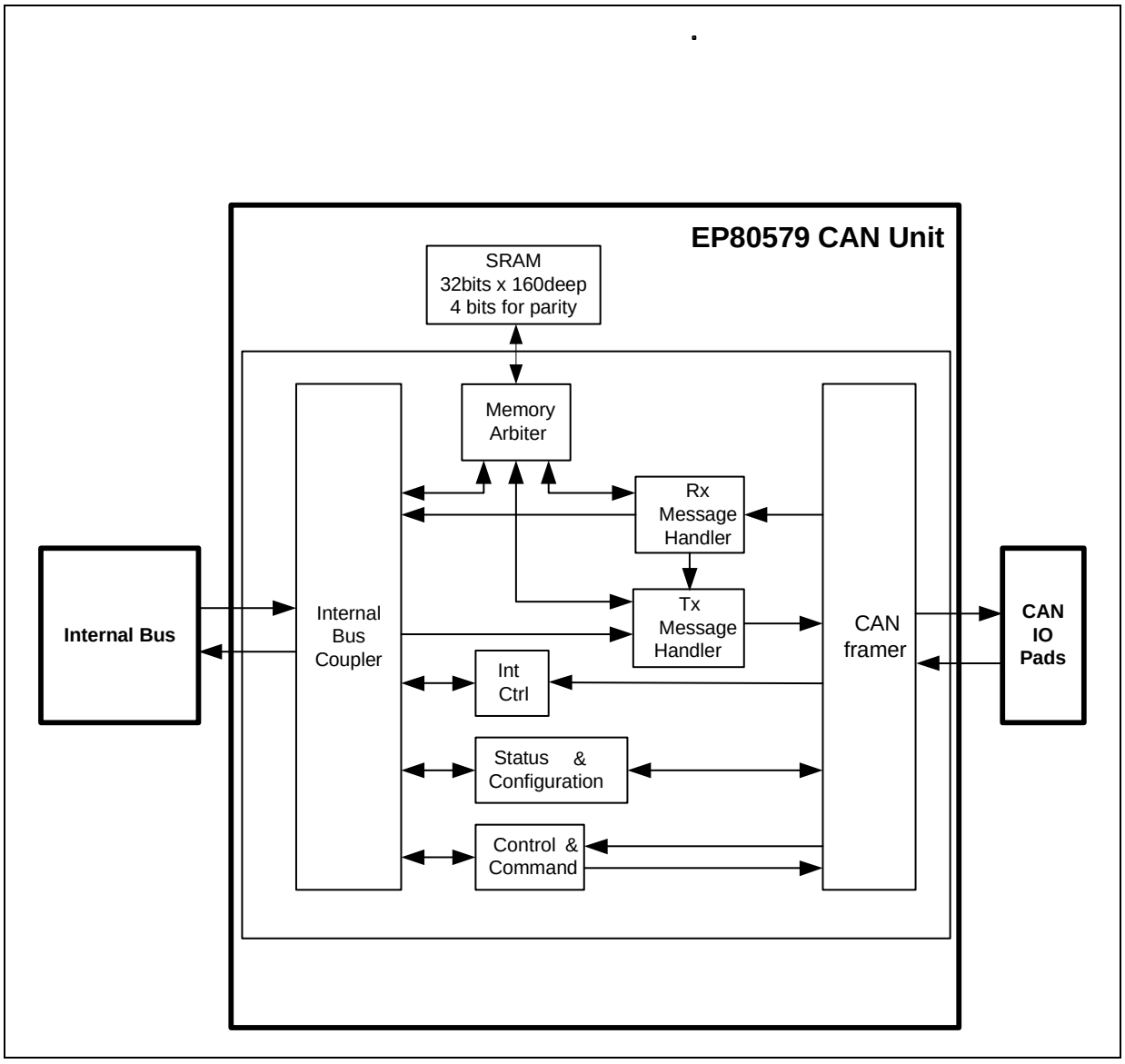
— Listen Only Mode.

- SRAM-based message buffer:
  - — 1Odd/Even parity generation and checking.
  - — Parity error reporting and parity test mode.

39.3 Functional Block Diagram

The block diagram for the CAN unit and its interface to other major blocks on the EP80579 is shown in Figure 39-1. The figure shows only one instance of the CAN unit. There are two instances of the same CAN unit.

Figure 39-1. CAN Block Diagram



39.4 Usage Model

39.4.1 CAN Basics

CAN is an asynchronous serial bus system. The bus structure is open and linear, with equal bus nodes. A CAN bus typically consists of two or more nodes. Nodes can be added to the CAN network dynamically without interrupting the communication of other nodes. That makes it easy to add or take off bus nodes (e.g. for adding functionality to the system, error recovery or monitoring).

The CAN bus is a “wired-AND” mechanism, where recessive bits (high) are overwritten by dominant bits (low). If no nodes are driving a dominant bit, the bus stays in the recessive state (idle). The node that first drives a dominant bit becomes the master and owns the bus. One of the cheapest and most common mediums is a twisted differential wire pair. The two lines are “CAN_H” and CAN_L”, which can be connected directly to the nodes via a connector. The maximum speed for the CAN bus is 1Mbps up to 40m. For bus length beyond 40 m, the bus speed must be reduced. Up to 30 nodes can be connected without extra equipment. CAN is insensitive to EMI because a differential pair will be affected in the same way.

The bit stream in a message is coded according on the Non-Return-to-Zero (NRZ) method, which means the signal level remains constant over the bit time and thus one time slot is required to represent a bit. To guarantee proper synchronization of all bus nodes, bit stuffing must be used. If five consecutive bits of the same polarity have been driven by the master, the sixth bit will be of opposite polarity, then the master will continue to transmit. Every receiver on the bus will check for the amount of bits with the same polarity, and will destuff (the sixth bit) the message.

39.4.2 Addressing and Bus Arbitration

The CAN protocol uses CSMA/CD with NDA (Carrier Sense Multiple Access/Collision Detection with Non-Destructive Arbitration) for bus arbitration.

A node that wants to transmit a message onto the network must first check if the CAN bus is in the “idle” state (Carrier Sense). The node then becomes the master by transmitting its message. Every other node will switch to receive mode during the Start of Frame bit, see Figure 39-2. After receiving a message with no errors, all nodes sends an acknowledge, and store the message if required. If message storage is not required, the message is discarded.

If multiple nodes start their transmission simultaneously (Multiple Access), collision is avoided by the use of bitwise arbitration (Collision Detection/Non-Destructive Arbitration), and the “Wired-AND” mechanism. The node that sends out a recessive bit in its identifier field (MSB first), but reads back a dominant bit, loses arbitration and switches to receive mode. The competitive node won arbitration because it had a higher priority. This way, nodes with higher priority do not have to waste time re-sending their message. The nodes that lose arbitration will automatically attempt to re-send their message after the bus becomes “idle”.

Note: Different nodes cannot have similar identifiers on the same network.

Standard CAN (CAN2.0A) has an 11 bit identifier field, see Figure 39-2, allowing a total of 2048 identifiers. This limitation has been improved by the Extended CAN (CAN2.0B), which has either a 11 and/or a 29 bit identifier field, resulting in a total of 536 million identifiers. See Figure 39-2.

39.4.3 Frame Types

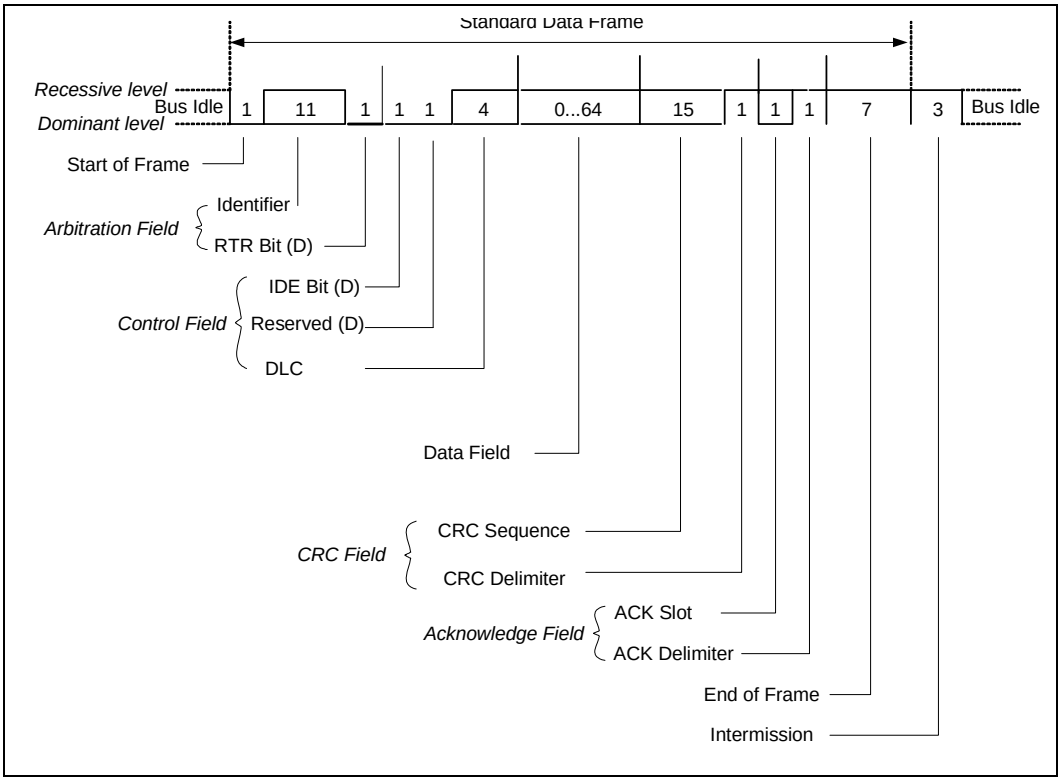
The CAN protocol is defined and controlled by four different frame types:

- Data Frame - Data frame from a transmitting node to receiving nodes.
- Remote Frame - Receiving node requesting the data frame with equal identifier.
- Error Frame - Transmitted by any node that detected an error.
- Overload Frame - A node can generate an overload frame under two conditions:
  - If the node detects a dominant bit during the interframe space (intermission).
  - The node is not yet ready to receive another data frame (wait states). Nodes are not allowed to generate more than 2 consecutive overload frames to delay the next transfer.
  - The CAN unit will not generate an overload frame but will respond to it if it sees one on the bus.

39.4.3.1 Data Frame

A standard CAN data frame is composed of seven fields, as shown in Figure 39-2.

Figure 39-2. Standard CAN Data Frame



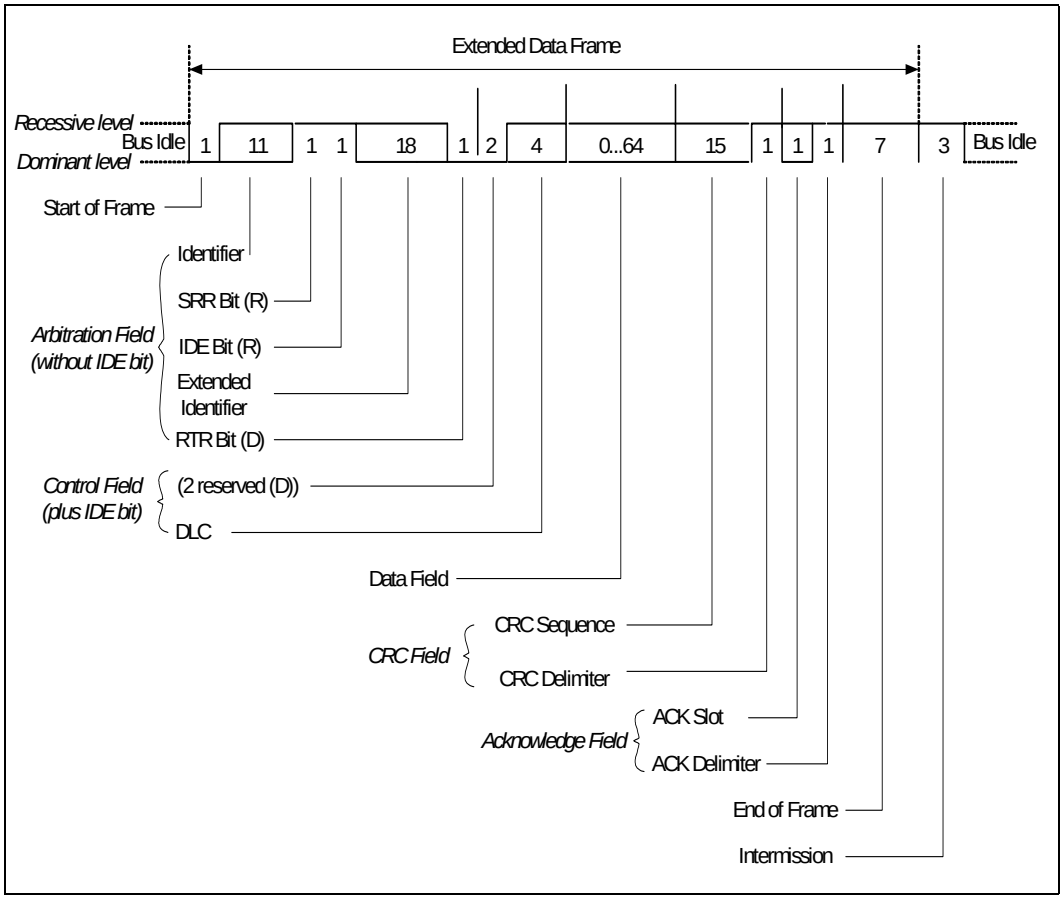
- Start of Frame: One dominant bit marks the beginning of Data Frames and Remote Frames. The purpose is hard synchronization for all nodes.
- Arbitration Field, consisting of:
  - 11 bit identifier
  - 1 RTR bit. This bit is used to distinguish between a Data Frame (Dominant) and a Remote Frame (Recessive).

- Control Field, consisting of:
  - — Data Length Code, which is 4 bits wide
  - — 2 reserved bits for future expansion
- Data Field, consisting of the data to be transmitted. The Data Field can be from 0 to 8 bytes. (MSB transferred first).
- CRC Field, consisting of:
  - — CRC Sequence, which is the remainder of a polynomial division. This is a 15 bit value.
  - — CRC delimiter which is one recessive bit.
- ACK Field, containing:
  - — a one bit ACK SLOT
    A transmitting node drives this bit with a recessive value. All receiving nodes with comparable CRC values acknowledge with a dominant bit.
  - — a one bit ACK Delimiter
    This will always be a recessive bit.
- End of Frame, which consists of seven recessive bits.

Between any frame the CAN bus must stay “idle” for at least three bit times(intermission). If no nodes wish to send a frame, the bus stays “idle”.

The Extended CAN Data Frame is illustrated in Figure 39-3.

Figure 39-3. Extended CAN Data Frame



Note: To to use both Extended and Standard Frames on the same network, it is necessary to split the 29 bit Extended identifier into one 11 bit (most significant) and one 18 bit (least significant) section. By doing this, the IDE bit can remain in the same bit position in both Standard and Extended Frames.

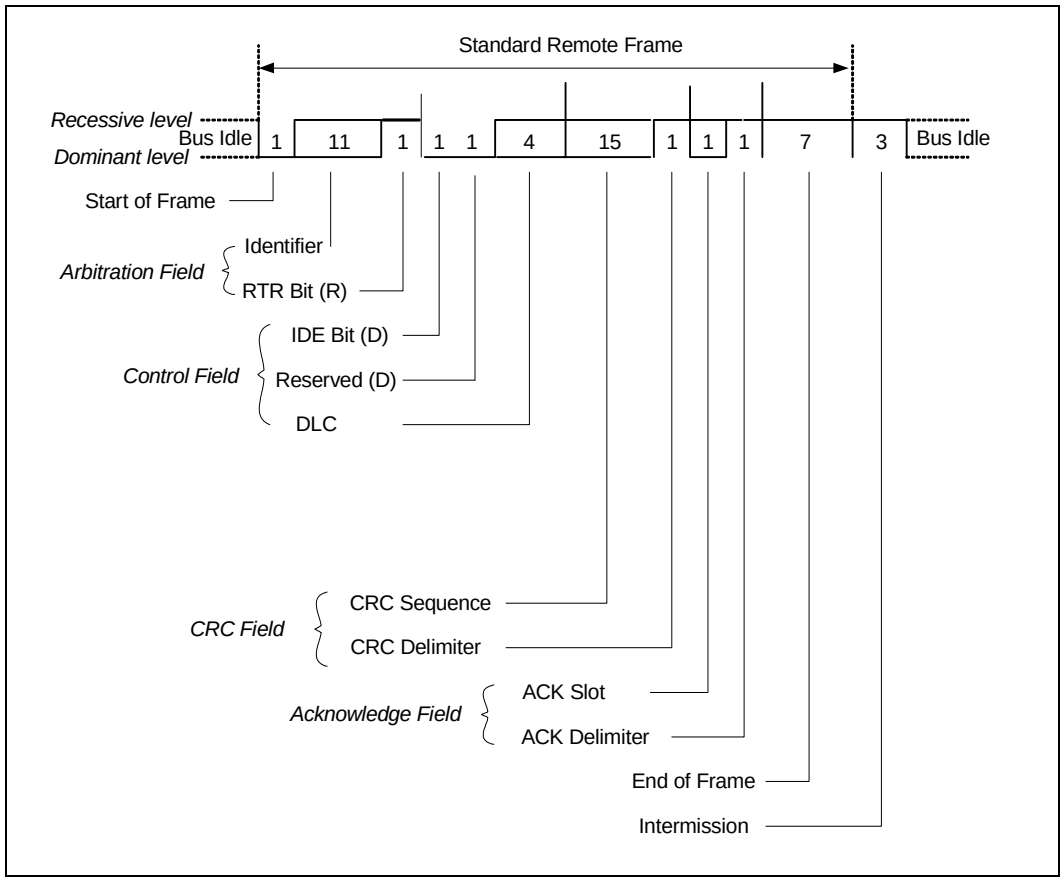
- Start of Frame: One dominant bit marks the beginning of both Data Frames and Remote Frames. This is for hard synchronization with all nodes.
- Arbitration Field, consisting of:
  - 11 bit identifier (most significant)
  - SRR bit, recessive
  - IDE bit, recessive to denote an Extended CAN Frame
  - Extended Identifier, 18 bits
  - 1 RTR bit. This bit is used to distinguish between a Data Frame(Dominant) and a Remote Frame (Recessive).
- Control Field, consisting of:
  - Two dominant bits
  - Data Length Code, which is 4 bits wide

The remaining fields are similar to Standard Data Frame.

39.4.3.2 Remote Frame

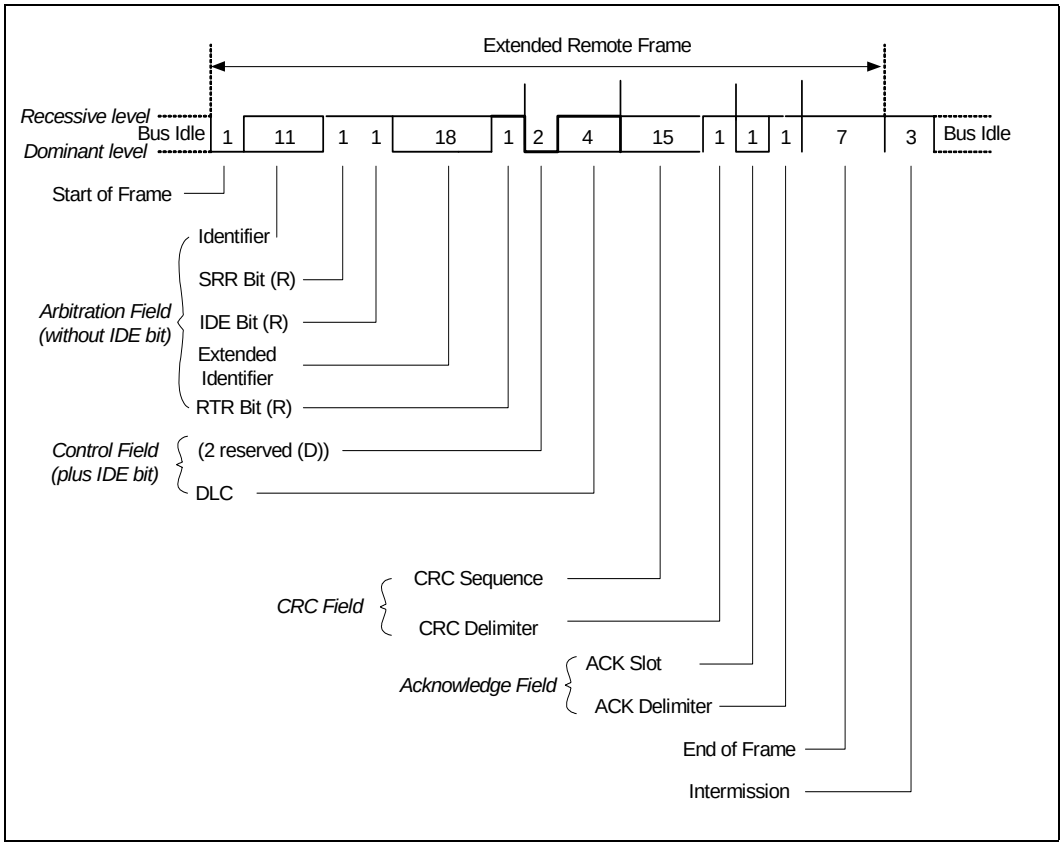
In the CAN protocol, it is also possible for a destination node to request data from a source. This can be done by transmitting a Remote Frame with an identifier that matches the identifier of the required Data Frame. This request results in a response from the source with the matching identifier, including the Data Frame. If a node were to transmit a Data Frame at the same time as a node were to send a Remote Frame with the same identifier, the Data Frame would win arbitration due to the dominant RTR bit.

Figure 39-4. Standard CAN Remote Frame



The format of an Extended Remote Frame is shown in Figure 39-5.

Figure 39-5. Extended CAN Remote Frame



39.4.3.3 Error Frames

CAN, as a message-oriented protocol, uses error signaling. Each network node checks each message that is transmitted on the bus for errors. When a transmitting or receiving network node detects an error, the node signals all other nodes by transmitting an error message (Error Frame). The Error Frame contains a six bit combination with the same polarity, normally as a dominant bit sequence (active error flag). All network nodes detect the error signal and cancel the segments of a message already received. Consistent data is thus ensured for all nodes of the network.

An Error Frame consists of two fields:

- Error Flag field:
  - An active Error Flag consists of six consecutive dominant bits. This violates the CAN protocol bit stuffing rule, and in turn all nodes on the network will generate error frames. The error frame field will then vary from a minimum of 6 to maximum of 12 consecutive dominant bits.
  - A passive Error Flag consists of six consecutive recessive bits, unless overwritten by dominant bits from other nodes. Unless the bus master transmits the passive Error Flag, the passive error flag will not affect any other node on the network.
- Error Delimiter field: consists of eight recessive bits and terminates the Error Frame.

When a transmitting node has transmitted an error frame to completion or has received one, it immediately attempts to transmit the previously transmitted message again with another bus arbitration process.

The error signaling mechanism ensures that the message transfer with all nodes of a network is error-free and consistent. Because error signaling takes place immediately after an error is detected, short error recovery times are guaranteed.

Note: The CAN protocol also provides a self-monitoring mechanism that prevents one dysfunctional node from holding the network down. The assessment of a node's statistical error rate may result in deactivation of the node.

39.4.3.4 Overload Frames

An Overload Frame has the same format as an active Error Frame. The difference is when the Overload Frame is transmitted and its purpose: An Overload Frame is only permitted to transmit during Interframe Space (Intermission). Two conditions lead to the generation of an Overload Frame:

- If a node is not yet ready for any data (insert wait state).
- A node detects a dominant bit during intermission. A node may generate at most 2 consecutive Overload Frames, to delay the next Data or Remote Frame.

Overload Frames consist of two fields:

- The Overload Flag has the same format as an active Error Frame.
- The Overload Delimiter consists of eight recessive bits.

Note: The CAN controller will not generate overload frames. However, it will respond to overload frames when it sees one on the bus.

39.4.4 CAN Bit Timing

39.4.4.1 Introduction

The CAN network is configured based on the bit rate. The nominal bit rate is the number of bits per second passing through the network in the absence of re synchronization. Each member of this network can implement the required timing parameters for a given bit rate using different values as long as the chosen values meet the synchronization requirements.

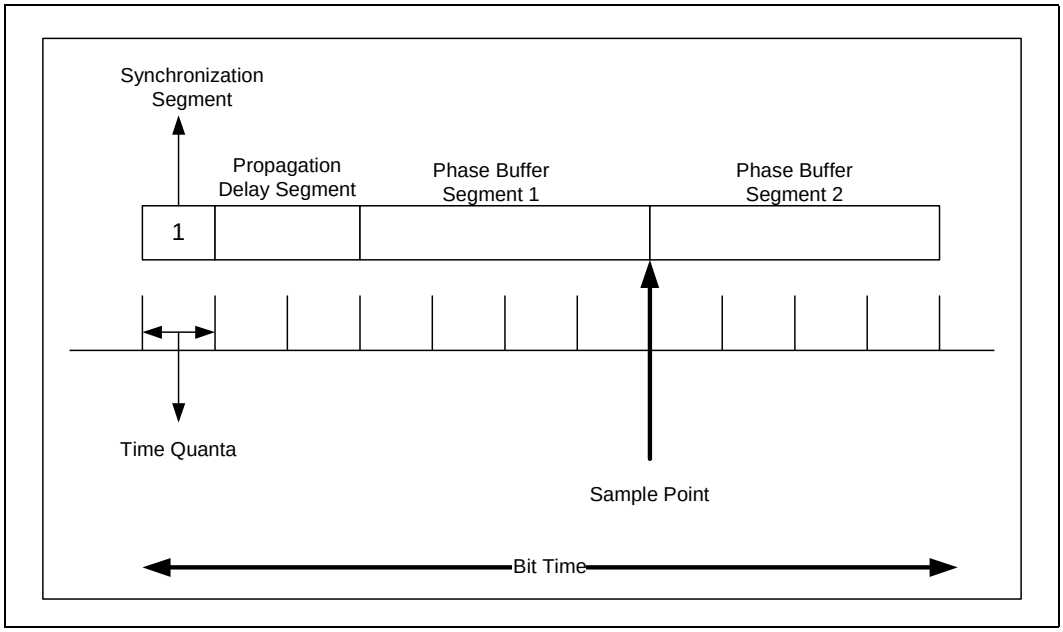
Oscillator drift and other issues could make the nominal configured bit rate to vary but the CAN protocol provides a mechanism to re synchronize the messages.

The nominal bit rate is made up of the following non-overlapping segments:

- Synchronization Segment: This is the start of the frame segment of the bit being transmitted. It is used primarily to synchronize nodes on the bus. The synchronization segment is always fixed to 1 time quanta.
- Propagation Segment: The is the time required for signals to travel in the CAN network. It includes the time required to travel from one end of the system to the other and back again. It also accounts for the CAN components to react.
- Phase Buffer Segment 1 & 2: These phase buffers exists to compensate for timing problems created by the oscillator drift and other issues. These 2 segments are adjusted in units called the Synchronization Jump Width.
- Sample Point: The sample point always occurs between the Phase 1 and Phase 2 segments. It can be set to be either single or multiple.

Each member of CAN network can use different timing parameters to achieve the same bit rate. As shown in Figure 39-6 each segment is defined in terms of time quantum (tq).

Figure 39-6. CAN Timing Parameters



The time quantum value is the basic unit of the bit time and is determined by the CAN controller system clock frequency and the bit scaler.

tq = Bit_Scaler / CAN clock frequency.

For a given bit rate that the CAN requires, the Bit_Scaler can be changed to meet the recommended values of phase buffer segments and synchronization jump width.

The CiA (CAN in automation) draft standard 102 version 2.0 defines the content of the physical layer and the basic characteristics of the physical medium, for communication according to the Controller Area Network protocol specification (CAN) between different types of electronic modules in general industrial applications.

Table 39-1. CiA Recommended bit rate and timing Parameters (Sheet 1 of 2)

Bit Rate (kbits/s)	Bus Length (m)	Nominal Bit Time tb (us)	Number of Time Quanta per Bit	Length of Time Quantum Tq (us)	Location of Sample Point wrt tq	Location of Sample Point (us)	Comments
1000	25	1	8	0.125	6 Tq	0.75	max 40m
800	50	1.25	10	0.125	8 Tq	1	
500	100	2	16	0.125	14 Tq	1.75	
250	250	4	16	0.250	14 Tq	3.5	
125	500	8	16	0.500	14 Tq	7	
50	1000	20	16	1.25	14 Tq	17.5	max 1km
20	2500	50	16	3.125	14 Tq	43.75	Has to be supported by all modules

Table 39-1. CiA Recommended bit rate and timing Parameters (Sheet 2 of 2)

Bit Rate (kbits/s)	Bus Length (m)	Nominal Bit Time tb (us)	Number of Time Quanta per Bit	Length of Time Quantum Tq (us)	Location of Sample Point wrt tq	Location of Sample Point (us)	Comments
10	5000	100	16	6.25	14 Tq	87.5	Minimum bit rate
Sampling Mode = Single Synchronization Mode = Reccessive to dominant edges only Synchronization Jump Width = 1 * Tq Phase Segment 2 = 2 * Tq							

The CiA recommended timing parameters for each bit rate is show in Figure 39-7.

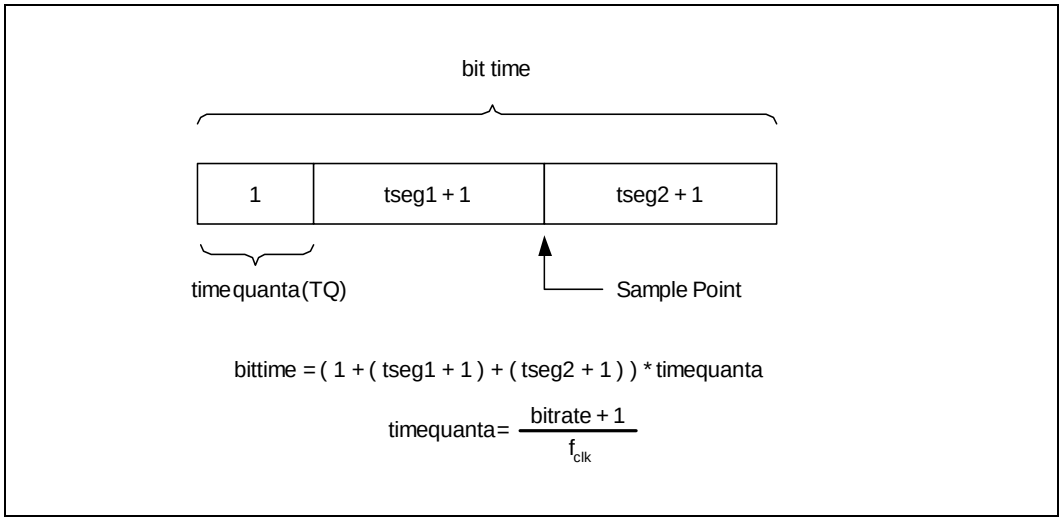
39.4.4.2 Setting Proper Bit Rate, tseg1 and tseg2

The CAN controller unit provides the following programmable timing parameters, which can be used to achieve CiA recommended bit rates:

- cfg_tseg1 (Propagation Delay Segment + Phase Buffer Segment 1): This is used to compensate for edge phase errors but also consists of a propagation segment, which is used to compensate for signal delays in the network.
- cfg_tseg2 (Phase Buffer Segment 2): This is also used for compensation of edge phase errors.
- cfg_sjw (Synchronization Jump Width): This defines how far the re-synchronization may move the sample point inside the limits defined by the phase buffer segments to compensate for edge phase errors.

As shown in Figure 39-7, the sample point separates tseg1 and tseg 2, this is the point in time where the bus level is read and interpreted.

Figure 39-7. Bit Rate and Time Settings



The CAN unit in the EP80579 operates at a frequency of 40Mhz. Based on this frequency, the recommended timing parameter values are as show in Table 39-2.

Table 39-2. CAN Recommended Bit Rate and Timing Parameters

Bit Rate (Kbs/s)	Nominal Bit Time (us)	Number of Time Quanta Per Bit	Nominal Bit Time / CAN Controller Frequency †	Time Quantum (us)	Bit Rate Scaler Required for 40Mhz CAN Clock	cfg_tseg1 (Tseg - 1)	cfg_bitrate
1000	1	8	40	0.125	5	4	4
800	1.25	10	50	0.125	5	6	4
500	2	16	80	0.125	5	12	4
250	4	16	160	0.250	10	12	9
125	8	16	320	0.500	20	12	19
50	20	16	800	1.250	50	12	49
20	50	16	2000	3.125	125	12	124
10	100	16	4000	6.250	250	12	249
The bit rate scaler is chosen such that this is an integer value. CAN Clock Frequency = 40Mhz cfg_tseg2 = 1 cfg_sjw = 0 Single Sampling Mode Synchronization Mode: Recessive to dominant only							

39.5 Theory of Operation

39.5.1 Modes of Operation

The CAN unit can be used in different operating modes. By disabling transmitting data, it is possible to use the CAN in listen-only mode, enabling features such as automatic bit rate detection. The two units can be used in an on-chip loop-back mode.

Before starting the CAN controller, all CAN configuration registers must be set according to the target application.

39.5.2 Error Handling

The CAN protocol provides sophisticated error detection mechanisms. The following list of errors can be detected:

- Cyclic Redundancy Check (CRC) Error - The received CRC sequence did not match the calculated result.
- Acknowledge Error - The transmitter did not detect a dominant bit during ACK slot.
- Form Error - One or more illegal bits in a fixed-form bit field.
- Bit Error - The monitored bit value is different from the bit value sent.
- Stuff Error - Between Start of Frame and CRC Delimiter, 6 consecutive bits with the same polarity are detected.

An interrupt will be generated if an error is enabled and detected, see “Offset 00000000h: Int_status - Interrupt Status Register”.

Note: The errors are always detected and registered in the interrupt status register. The interrupt enable register determines whether or not it gets propagated. For more information please refer to "Offset 00000000h: Int_status - Interrupt Status Register" and "Offset 00000004h: Int_Ebl - Interrupt Enable Register"

39.5.3 Send/Receive Procedure

39.5.3.1 Send Procedure

Use the following procedure to send a message:

1. Write a message into one of the transmit message holding buffers. An empty buffer is indicated by a TxReq that equals zero.
2. Request transmission by setting the respective TxReq flag to '1'.
3. The TxReq flag remains set as long as the message transmit request is pending. The content of the message buffer must not change while the TxReq flag is set.
4. The internal message priority arbiter selects the message according to the chosen arbitration scheme.
5. Once the message is transmitted, the TXReq flag is set to zero and the TxMsg interrupt status is asserted.

A message can be removed from a transmit holding buffer by setting the TxAbort flag. Use the following procedure to remove the contents of a particular TxMessage buffer:

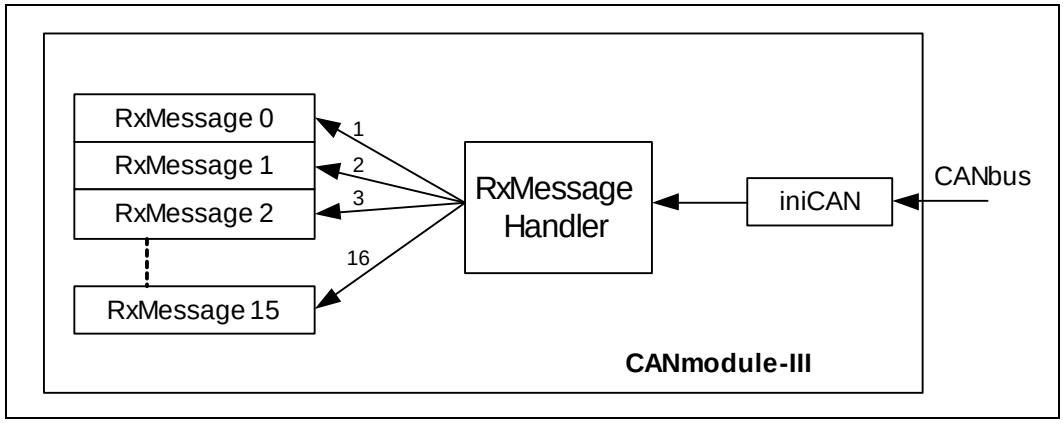
1. Set TxAbort to '1' to request the message removal.
2. The TxAbort flag remains set as long as the message abort request is pending. The flag is cleared when either the message wins arbitration (TxMsg interrupt active) or the message is removed (TxMsg interrupt inactive).

39.5.3.2 Receive Procedures

The CAN unit supports 16 individual receive message buffers. Each one has its own message filter mask. Automatic reply to RTR messages is supported.

If a message is accepted in a receive buffer, its MsgAv flag is set. The message remains valid as long as MsgAv flag is set. The host CPU has to reset the MsgAv flag to enable receipt of a new message.

Figure 39-8. Receive Message Handler



39.5.3.3 Rx Message Processing

After receipt of a new message, the RxMessageHandler searches all receive buffers starting from RxMessage0 until it finds a valid buffer.

A valid buffer is indicated by:

- Receive buffer is enabled indicated by RxBufferEbl = '1'.
- Acceptance Filter of receive buffer matches incoming message.

If the RxMessageHandler finds a valid buffer that is empty, then the message is stored and the MsgAv flag is set to '1'. If the RxIntEbl flag is set, then the RxMsg flag of the interrupt controller is set. If the receive buffer already contains a message indicated by MsgAv = '1' and the Link Flag is not set, then a RxMsgLoss interrupt flag is set.

If an incoming message has its RTR flag set and the RTRreply flag is set, then the message is not stored but an RTR auto-reply is issued. See "Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control" for more details.

39.5.3.4 Acceptance Filter

Each receive buffer has its own acceptance filter that is used to filter incoming messages. An acceptance filter consists of an Acceptance Mask Register (AMR) and Acceptance Code Register (ACR) pair. The AMR defines which bits of the incoming CAN message have to match the respective ACR bits.

The following message fields are covered:

- ID
- IDE
- RTR
- Data byte 1 and data byte 2 (DATA[63:56])

Note: Some CAN High Level Protocols such as SDS or Device Net carry additional protocol related information in the first or first two data bytes that are used for message acceptance and selection. The capability to filter on these fields provides a more efficient implementation of the protocol stack running on the CPU.

The acceptance mask register (AMR) defines whether the incoming bit is checked against the acceptance code register (ACR).

- AMR:
  - '0': The incoming bit is checked against the respective ACR. The message is not accepted when the incoming bit doesn't match the respective ACR flag.
  - '1': The incoming bit is 'don't care'.

Example:

The following example shows the acceptance register settings used to support receipt of a CANopen TPD01 (Transmit Process Data Object) message. In CANopen, a widely used CAN Higher Level Protocol (HLP), the ID bits are used to select the message type. The bit assignment is shown in the following table:

Table 39-3. CAN Higher Level Protocol (HLP) Bit Assignment

CANopen Identifier										
10	9	8	7	6	5	4	3	2	1	0
Function Code					Node-ID					

Identifier fields:

- Function Code: The function code for a TDPO1 message is 3h.
- Node-ID: The example uses 02h as the Node ID.
- IDE = ‘0’: CANopen uses the short format message.
- RTR = ‘0’:Identifies a regular message.

To accept this message, the acceptance filter settings would look like the following:

- AMR settings:
  ID[28:18] = 0
  ID[17:0] = all ones
  IDE = 0
  RTR = 0
  DATA[63:56] = all ones
- ACR settings:
  ID[28:18] = 182h
  ID[17:0] = don't care
  IDE = 0
  RTR = 0
  DATA[63:56] = don’t care

39.5.3.5 RTR Auto-Reply

The CAN unit supports fully automatic answering of RTR message requests. All 16 receive buffers support this feature. If an RTR message is accepted in a receive buffer, where the RTRreply flag is set, then this buffer automatically replies to this message with the content of this receive buffer. The RTRreply_pending flag is set when the RTR message request is received. The flag is reset when the message was sent or when the message buffer is disabled. To abort a pending RTRreply message, use the RTRabort command.

39.5.3.6 RxBuffer Linking

Several receive buffers may be linked together to form a receive buffer array which acts like a receiver FIFO.

The requirements are as follows:

- All buffers of the same array must have the same message filter setting (AMR and ACR are identical).
- The last buffer of an array may not have its link flag set.

When a receive buffer already contains a message (MsgAv=’1’) and a new message arrives for this buffer, then this message would be discarded (RxMsgLoss Interrupt). To avoid this situation, several receive buffers can be linked together. When the CAN unit receives a new message, the RxMessage handler searches for a valid receive buffer. If

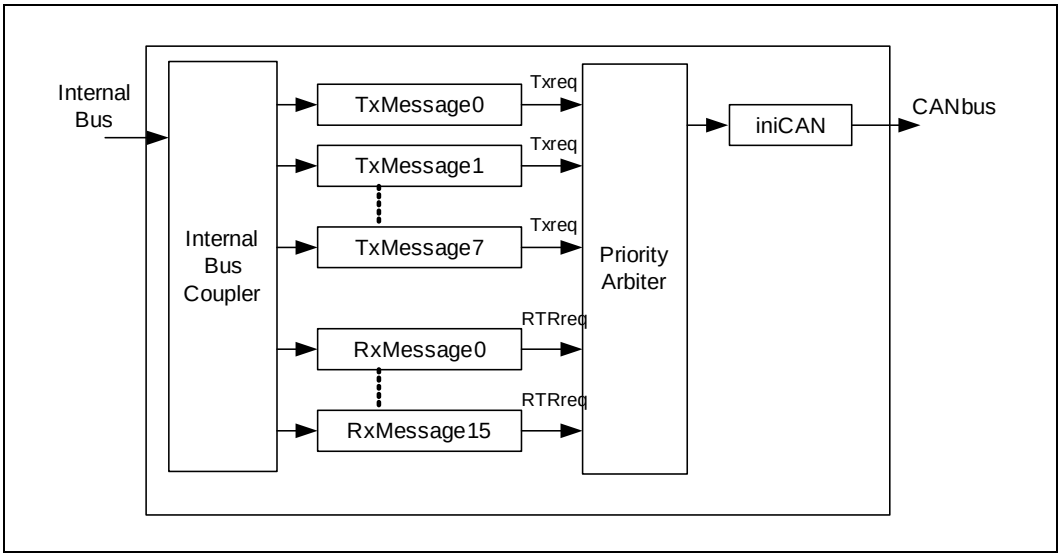
one is found that is already full (MsgAv='1') and the link flag is set (BufferLink='1'), the search for a valid receive buffer is continued. If no other buffer is found, then the RxMsgLoss interrupt is set anyway.

It is possible to build several message arrays. Each of these arrays must use the same AMR and ACR.

39.5.4 TxMessage Registers

Eight transmit message holding buffers are provided. An internal priority arbiter selects the message according to the chosen arbitration scheme. Upon transmission of a message or message arbitration loss, the priority arbiter re-evaluates the message priority of the next message.

Figure 39-9. Message Arbitration



The priority arbiter supports round robin and fixed priority arbitration. The arbitration mode is selected using the configuration register.

- Round Robin - Buffers are served in a defined order: 0-1-2..7-0-1... A particular buffer is only selected if its TxReq flag is set. This scheme guarantees that all buffers receive the same probability to send a message.
- Fixed priority - Buffer 0 has the highest priority. With this mode, it is possible to designate buffer 0 as the buffer for error messages and it is guaranteed that they are sent first.

Note: RTR message requests are served before TxMessage buffers are handled (e.g. RTRreq0, ... RTRreq15, TxMessage0, TxMessage0, TxMessage1, ... TxMessage7).

See the description of the configuration register in the TxMessage0 Buffer in "Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command" for information about how to select these modes.

39.6 Register Summary

The default value for all registers is 0h, unless otherwise noted. All registers starting at 20h are implemented in SRAM without guaranteed reset values. It is the responsibility of either BIOS or SW to initialize these to 0.

For more information on the conventions the following register summaries adopt, see Section 7.1, "Overview of Register Descriptions and Summaries" on page 183.

The CAN registers materialize in the PCIspace.

Table 39-4 and Table 39-5 summarize the CAN interface #0 and #1 materializations from the PCI perspective.

Table 39-4. Bus M, Device 4, Function 0: Summary of CAN Registers Mapped Through CSRBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 00000000h: Int_Status - Interrupt Status Register" on page 1587	00000000h
00000004h	00000007h	"Offset 00000004h: Int_Ebl - Interrupt Enable Register" on page 1588	00000000h
00000008h	0000000Ah	"Offset 00000008h: Buffer Status Indicators" on page 1589	00000000h
0000000Ch	0000000Fh	"Offset 0000000Ch: ErrorStatus - Error Status Indicators" on page 1590	00000000h
00000010h	00000013h	"Offset 00000010h: Command - Operating Modes" on page 1591	00000000h
00000014h	00000017h	"Offset 00000014h: Config - CAN Configuration Register" on page 1592	00000000h
00000020h at 10h	00000023h at 10h	"Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command" on page 1593	XXXXXXXXh
00000024h at 10h	00000027h at 10h	"Offset 00000024h: TxMessageID[0-7] - Transmit Message ID" on page 1595	XXXXXXXXh
00000028h at 10h	0000002Ah at 10h	"Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High" on page 1596	XXXXXXXXh
0000002Ch at 10h	0000002Fh at 10h	"Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low" on page 1597	XXXXXXXXh
000000A0h at 20h	000000A3h at 20h	"Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control" on page 1598	XXXXXXXXh
000000A4h at 20h	000000A7h at 20h	"Offset 000000A4h: RxMessageID[0-15] - Receive Message ID" on page 1600	XXXXXXXXh
000000A8h at 20h	000000ABh at 20h	"Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High" on page 1600	XXXXXXXXh
000000ACh at 20h	000000AFh at 20h	"Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low" on page 1601	XXXXXXXXh
000000B0h at 20h	000000B3h at 20h	"Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR" on page 1601	XXXXXXXh
000000B4h at 20h	000000B7h at 20h	"Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR" on page 1602	XXXXXXXXh
000000B8h at 20h	000000BBh at 20h	"Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data" on page 1603	XXXXXXXXh
000000BCh at 20h	000000BFh at 20h	"Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data" on page 1604	XXXXXXXXh

Table 39-5. Bus M, Device 5, Function 0: Summary of CAN Registers Mapped Through CSRBAR Memory BAR

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 00000000h: Int_Status - Interrupt Status Register" on page 1587	00000000h
00000004h	00000007h	"Offset 00000004h: Int_Ebl - Interrupt Enable Register" on page 1588	00000000h
00000008h	0000000Ah	"Offset 00000008h: Buffer Status Indicators" on page 1589	00000000h
0000000Ch	0000000Fh	"Offset 0000000Ch: ErrorStatus - Error Status Indicators" on page 1590	00000000h
00000010h	00000013h	"Offset 00000010h: Command - Operating Modes" on page 1591	00000000h
00000014h	00000017h	"Offset 00000014h: Config - CAN Configuration Register" on page 1592	00000000h
00000020h at 10h	00000023h at 10h	"Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command" on page 1593	XXXXXXXXh
00000024h at 10h	00000027h at 10h	"Offset 00000024h: TxMessageID[0-7] - Transmit Message ID" on page 1595	XXXXXXXXh
00000028h at 10h	0000002Ah at 10h	"Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High" on page 1596	XXXXXXXXh
0000002Ch at 10h	0000002Fh at 10h	"Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low" on page 1597	XXXXXXXXh
000000A0h at 20h	000000A3h at 20h	"Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control" on page 1598	XXXXXXXXh
000000A4h at 20h	000000A7h at 20h	"Offset 000000A4h: RxMessageID[0-15] - Receive Message ID" on page 1600	XXXXXXXXh
000000A8h at 20h	000000ABh at 20h	"Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High" on page 1600	XXXXXXXXh
000000ACh at 20h	000000AFh at 20h	"Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low" on page 1601	XXXXXXXXh
000000B0h at 20h	000000B3h at 20h	"Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR" on page 1601	XXXXXXXh
000000B4h at 20h	000000B7h at 20h	"Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR" on page 1602	XXXXXXXXh
000000B8h at 20h	000000BBh at 20h	"Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data" on page 1603	XXXXXXXXh
000000BCh at 20h	000000BFh at 20h	"Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data" on page 1604	XXXXXXXXh

39.6.1 Detailed Register Descriptions

This section shows all internal registers and describes how the CAN unit can be used and programmed.

39.6.1.1 Offset 00000000h: Int_status - Interrupt Status Register

The interrupt controller contains an interrupt status and an interrupt enable register. The interrupt status register stores internal interrupt events. Once a bit is set, it remains set until it is cleared by writing a '1' to it. The interrupt enable register has no effect on the interrupt status register.

The interrupt enable register controls which particular bits from the interrupt status register are used to assert the interrupt output int_n. int_n is asserted if a particular interrupt status bit and the respective enable bit are set.

Table 39-6. Offset 00000000h: Int_Status - Interrupt Status Register

Description:					
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:4:0		**Offset Start:** 00000000h **Offset End:** 00000003h	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:5:0		**Offset Start:** 00000000h **Offset End:** 00000003h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :13	Reserved	Reserved	N	0h	RO
12	rx_msg	Indicates that a message was received.	N	0b	RWC
11	tx_msg	Indicates that a message was sent.	N	0b	RWC
10	rx_msg_loss	Is set when a new message arrives but the RxMessage flag MsgAv is set	N	0b	RWC
09	bus_off	The CAN has reached the bus off state	N	0b	RWC
08	crc_err	An crc error occured while receiving or transmitting data	N	0b	RWC
07	form_err	A form error occured while receiving or transmitting data.	N	0b	RWC
06	ack_err	An acknowledge error occured while transmitting data	N	0b	RWC
05	stuff_err	A stuff error occured while transmitting data	N	0b	RWC
04	bit_err	A bit error occured while receiving or transmitting data	N	0b	RWC
03	ovr_load	An overload condition has occured	N	0b	RWC
02	ar_loss	The arbitration was lost while sending a message	N	0b	RWC
01 :00	Reserved	Reserved	N	0h	RV

39.6.1.2 Offset 00000004h: Int_Ebl - Interrupt Enable Register

Table 39-7. Offset 00000004h: Int_Ebl - Interrupt Enable Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000004h **Offset End:** 00000007h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000004h **Offset End:** 00000007h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :13	RSVD	Reserved, these bits are always 0		0h	RW
12	rx_msg	Indicates that a message was received.		0h	RW
11	tx_msg	Indicates that a message was sent.		0b	RW
10	rx_msg_loss	Is set when a new message arrives but the RxMessage flag MsgAv is set		0b	RW
09	bus_off	The CAN has reached the bus off state		0b	RW
08	crc_err	A CRC error occured while receiving or transmitting data		0b	RW
07	form_err	A form error occured while receiving or transmitting data.		0b	RW
06	ack_err	An acknowledge error occured while transmitting data		0b	RW
05	stuff_err	A stuff error occured while transmitting data		0b	RW
04	bit_err	A bit error occured while receiving or transmitting data		0b	RW
03	ovr_load	An overload condition has occured		0b	RW
02	ar_loss	The arbitration was lost while sending a message		0b	RW
01	RSVD	Reserved		0b	RW
00	int_ebl	int_ebl, global interrupt enable flag. 0 = All interrupts are disabled 1 = Enabled interrupt sources are available		0b	RW

39.6.1.3 Offset 00000008h: Buffer Status - Buffer Status Indicators

These status indicators bundle the respective flags from all RxMessage and TxMessage buffers.

Table 39-8. Offset 00000008h: Buffer Status Indicators

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000008h **Offset End:** 0000000Ah	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000008h **Offset End:** 0000000Ah	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RSVD	Reserved, these bits are always 0		0h	RO
23 :16	TXMSG_7_0	TxReq pending (bits [7:0])		0b	RO
15 :00	RXMSG_15_0	MsgAv (bits [15:0])		0b	RO

Note: All flags are read only (e.g. to acknowledge a MsgAv flag, the CPU has to write to the respective RxMessage buffer).

39.6.1.4 Offset 0000000Ch: ErrorStatus - Error Status Indicators

Status indicators are provided to report the CAN controller error state, receive error count and transmit error count. Special flags to report error counter values equal to or in excess of 96 errors are available to indicate heavily disturbed bus situations.

Table 39-9. Offset 0000000Ch: ErrorStatus - Error Status Indicators

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 0000000Ch **Offset End:** 0000000Fh	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 0000000Ch **Offset End:** 0000000Fh	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :20	RSVD	Reserved, these bits are always 0		0h	RO
19	rxgte96	The receiver error counter is greater or equal 96(dec)		0b	RO
18	txgte96	The transmitter error counter is greater or equal 96(dec)		0b	RO
17 :16	error_stat_1_0	The error state of the CAN node (bits[1:0]): “00” = error active (normal operation) “01” = error passive “1x” = bus off		0h	RO
15 :8	rx_err_cnt_7_0	The receiver error counter (bits [7:0]) according to the Bosch CAN 2.0 specification. When in bus off (inactive), this counter is used to count the idle states		0b	RO
07 :00	tx_err_cnt_7_0	The transmitter error counter (bits [7:0]) according to the Bosch* CAN 2.0 specification. When it is greater than 255(dec), it is fixed at 255.		0b	RO

39.6.1.5 Offset 00000010h: Command - Operating Modes

The CAN unit can be used in different operating modes. By disabling transmitting data, it is possible to use the CAN in listen-only mode. This allows for features such as automatic bit rate detection.

Before starting the CAN controller, all CAN controller registers have to be set according to the target application.

Table 39-10. Offset 00000010h: Command - Operating Modes

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000010h **Offset End:** 00000013h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000010h **Offset End:** 00000013h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :02	RSVD	Reserved, these bits are always 0		0h	RO
01	listen	Listen mode: '0' = Active '1' = CAN listen only: The output is held at 'R' level. The CAN is only listening		0b	RW
00	run_stop	Run/Stop mode: '0' = Sets the CAN controller into stop mode. Read '0' when stopped. '1' = Sets the CAN controller into run mode. Read '1' when running		0b	RW

Note: The run/stop bit [00] can not be written from RUN mode to STOP mode while a message is pending in the TX buffer. The stalled packet can be dequeued first using the ABORT bit in the buffer prior to setting the mode to STOP. If the requirement is to retain the TX packet in the buffer, transitioning to LISTEN mode first by setting bit [1] will allow the controller to then transition to STOP mode. The packet will then remain in the buffer if the controller is transitioned back to RUN mode.

39.6.1.6 Offset 00000014h: Config - CAN Configuration Register

The CAN unit must be configured prior to its use. The following registers define the effective CAN data rate, CAN data synchronization, and message buffer arbitration.

Note: Additional information on the CAN data rate settings using time segment1(tseg1), time segment2(tseg2), and the bit rate are given in Section 39.4.4, "CAN Bit Timing" on page 1577.

Table 39-11. Offset 00000014h: Config - CAN Configuration Register

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000014h **Offset End:** 00000017h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000014h **Offset End:** 00000017h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved, these bits are always 0		0b	RO
30 :16	cfg_bitrate_14_0	Prescaler (bits[14:0]) for generating the time quantum which defines the TQ: '0': One time quantum equals 1 clock cycle '1': One time quantum equals 2 clock cycles 32767: One time quantum equals 32768 clock cycles		0h	RW
15 :13	RSVD	Reserved.		0h	RW
12	cfg_arbiter	Transmit Buffer Arbiter '0': Round Robin: TxMessage0-1-2-3 etc. '1': Fixed Priority: TxMessage0 is highest, TxMessage2 is lowest		0b	RW
11 :08	cfg_tseg1	Time segment 1. Length of the first time segment: tseg1 = cfg_tseg1 +1 Time segment 1 includes the propagation time. Cfg_tseg1 = 0 and cfg_tseg1 =1 are not allowed.		0h	RW
7 :05	cfg_tseg2	Time segment 2. Length of the second time segment: tseg2 = cfg_tseg2 +1 Time segment 2 includes the propagation time. Cfg_tseg1 = 0 and cfg_tseg1 =1 are not allowed.		0h	RW
4	auto_restart	'0': After bus off, the CAN must be started 'by hand'. '1': After bus off, the CAN is restarting automatically after 128 groups of 11 recessive bits		0b	RW
03 :02	cfg_sjw	Synchronization jump width - 1 sjw <= tseg1 and sjw <= tseg2		0h	RW
1	sampling_mode	CAN bus bit sampling '0': One sampling point is used in the receiver path '1': 3 sampling points with majority decision are used		0b	RW
0	edge_mode	CAN bus synchronization logic '0': Edge from 'R' to 'D' is used for synchronization '1': Both edges are used		0b	RW

39.6.1.7 Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-12. Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000020h at 10h **Offset End:** 00000023h at 10h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000020h at 10h **Offset End:** 00000023h at 10h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RSVD	Reserved.		Xh	RW
23	WPN_21_16	Write Protect Not. Using the WPN flag enables simple retransmission of the same message by only having to set the TRX flag without taking care of the special flags. '0': Bit [21:16] remain unchanged. '1': Bit [21:16] are modified, default. NOTE: The readback state of this bit is undefined.		Xh	RW
22	RSVD	Reserved.		Xh	RW
21	RTR	Remote Bit '0': This is a standard message '1': This is an RTR message		Xh	RW
20	IDE	Extended identifier bit. '0': This is a standard format message '1': This is an extended format message		Xh	RW
19 :16	DLC	Data Length Code. Invalid values are transmitted as they are, but only the number of data bytes is limited to eight. 0: Message has 0 data byte, data[63:0] is not used 1: Message has 1 data byte, data [63:56] is used .... 8: Message has 8 data bytes, data [63:0] is used 9-15: Message has 8 data bytes		Xh	RW
15 :04	RSVD	Reserved.		Xh	RW
03	WPN_2	Message Control: WPN: Write Protect Not '0': Bit [2] remain unchanged. '1': Bit [2] is modified, default. NOTE: The readback state of this bit is undefined.		Xh	RW

Table 39-12. Offset 00000020h: TxMessageControl[0-7] - Transmit Message Control and Command (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000020h at 10h **Offset End:** 00000023h at 10h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000020h at 10h **Offset End:** 00000023h at 10h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	TxIntEbl	Tx Interrupt Enable '0': Interrupt disabled. '1': Interrupt enabled, successful message transmission set the TxMsg flag in the interrupt controller.		Xh	RW
01	TxAbort	Transmit Abort Request '0': Idle '1': Requests removal of a pending message. The message is removed the next time an arbitration loss happened. The flag is cleared when the message was removed or when the message won arbitration. The TxReq flag is released at the same time.		Xh	RW
00	TxReq	Transmit Request Write: '0': Idle. '1': Message Transmit Request. The Tx message buffer must not be changed while TxReq is '1'. Read: '0': TxReq completed. '1': TxReq pending.		Xh	RW

39.6.1.8 Offset 00000024h: TxMessageID[0-7] - Transmit Message ID

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-13. Offset 00000024h: TxMessageID[0-7] - Transmit Message ID

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 00000024h at 10h **Offset End:** 00000027h at 10h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 00000024h at 10h **Offset End:** 00000027h at 10h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :03	ID_28_0	Identifier (bits [28:0])		Xh	RW
02 :00	RSVD	Reserved.		Xh	RW

39.6.1.9 Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-14. Offset 00000028h: TxMessageDataHigh[0-7] - Transmit Message Data High

Description:					
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:4:0		**Offset Start:** 00000028h at 10h **Offset End:** 0000002Ah at 10h	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:5:0		**Offset Start:** 00000028h at 10h **Offset End:** 0000002Ah at 10h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DataHigh	TxMessage0 Buffer. Data [63:32]. Byte 1 is Data[63:56], Byte 2 is Data[55:48] etc.		Xh	RW

39.6.1.10 Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-15. Offset 0000002Ch: TxMessageDataLow[0-7] - Transmit Message Data Low

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 0000002Ch at 10h **Offset End:** 0000002Fh at 10h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 0000002Ch at 10h **Offset End:** 0000002Fh at 10h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	DataLow	TxMessage0 Buffer. Data [31:0].		Xh	RW

39.6.1.11 Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-16. Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control (Sheet 1 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR	**Bus:Device:Function:** M:4:0		**Offset Start:** 000000A0h at 20h **Offset End:** 000000A3h at 20h	
View: PCI 2	**BAR:** CSRBAR	**Bus:Device:Function:** M:5:0		**Offset Start:** 000000A0h at 20h **Offset End:** 000000A3h at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RSVD	Reserved.		Xh	RW
23	WPN_21_16	RxMessage Control: Write Protect Not High. '0': [21:16] remain unchanged '1': [21:16] are modified, default. This bit is always zero for readback		Xh	RW
22	RSVD	Reserved.		Xh	RW
21	RTR	Remote Bit		Xh	RW
20	IDE	Extended Identifier Bit		Xh	RW
19 :16	DLC	Data Length Code. Invalid values are shown as received.		Xh	RW
15 :08	RSVD	Reserved.		Xh	RW
07	WPN_6_3	RxMessage Control: Write Protect Low '0': [6:3] remain unchanged '1': [6:3] are modified, default. This bit is always zero for readback		Xh	RW
06	LF	Link Flag '0': This buffer is not linked, or it is the last one of an array '1': This buffer is linked with other buffers		Xh	RW
05	RxIntEbl	Receive Interrupt Enable '0': Interrupt is disabled '1': Interrupt is enabled		Xh	RW
04	RTRreply	Automatic message reply to RTR messages '0': automatic RTR disabled '1': automatic RTR enabled		Xh	RW
03	BE	Buffer Enable '0': Buffer is disabled '1': Buffer is enabled		Xh	RW

Table 39-16. Offset 000000A0h: RxMessageControl[0-15] - Receive Message Command and Control (Sheet 2 of 2)

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000A0h at 20h **Offset End:** 000000A3h at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000A0h at 20h **Offset End:** 000000A3h at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	RTRabort	RTR Abort request '0': Idle '1': Request removal of a pending RTR message reply. The flag is cleared when the message was removed or when the message won arbitration. The TxReq flag is released at the same time		Xh	RW
01	RTRreply_pending	Reply Pending '0': No RTR reply request pending '1': RTR reply request pending		Xh	RW
00	MsgAv	Message Available Read: '0': No new message available 1': New message available Write: '0': idle '1': Acknowledges receipt of new message. Acknowledging a message clears the MsgAv flag. Before acknowledging receipt of a new message, the message content must be copied into system memory. Acknowledging a message clears the MsgAv flag.		Xh	RW

39.6.1.12 Offset 000000A4h: RxMessageID[0-15] - Receive Message ID

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-17. Offset 000000A4h: RxMessageID[0-15] - Receive Message ID

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000A4h at 20h **Offset End:** 000000A7h at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000A4h at 20h **Offset End:** 000000A7h at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :03	RX_ID_28_0	Identifier (bits [28:0])		Xh	RW
02 :00	RSVD	Reserved		Xh	RW

39.6.1.13 Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-18. Offset 000000A8h: RxMessageDataHigh[0-15] - Receive Message Data High

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000A8h at 20h **Offset End:** 000000ABh at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000A8h at 20h **Offset End:** 000000ABh at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	RxDataHigh	RxMessage0 Buffer. Data [63:32].		Xh	RW

39.6.1.14 Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-19. Offset 000000ACh: RxMessageDataLow[0-15] - Receive Message Data Low

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000ACh at 20h **Offset End:** 000000AFh at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000ACh at 20h **Offset End:** 000000AFh at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :00	RxDataLow	RxMessage0 Buffer. Data [31:0].		Xh	RW

39.6.1.15 Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-20. Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000B0h at 20h **Offset End:** 000000B3h at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000B0h at 20h **Offset End:** 000000B3h at 20h	
Size: 32 bit	**Default:** XXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :03	Identifier	Identifier		Xh	RW

Table 39-20. Offset 000000B0h: RxMessageAMR[0-15] - Receive Message AMR

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:4:0	Offset Start: 000000B0h at 20h Offset End: 000000B3h at 20h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:5:0	Offset Start: 000000B0h at 20h Offset End: 000000B3h at 20h	
Size: 32 bit	Default: XXXXXXXh			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
02	IDE	Extended identifier bit		Xh	RW
01	RTR	Remote bit		Xh	RW
00	RSVD	Reserved		Xh	RW

39.6.1.16 Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-21. Offset 000000B4h: RxMessageACR[0-15] - Receive Message ACR

Description:					
View: PCI 1	BAR: CSRBAR		Bus:Device:Function: M:4:0	Offset Start: 000000B4h at 20h Offset End: 000000B7h at 20h	
View: PCI 2	BAR: CSRBAR		Bus:Device:Function: M:5:0	Offset Start: 000000B4h at 20h Offset End: 000000B7h at 20h	
Size: 32 bit	Default: XXXXXXXXh			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :03	Identifier	Identifier		Xh	RW
02	IDE	Extended identifier bit		Xh	RW
01	RTR	Remote bit		Xh	RW
00	RSVD	Reserved		Xh	RW

39.6.1.17 Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-22. Offset 000000B8h: RxMessageAMR_Data[0-15] - Receive Message AMR Data

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0		**Offset Start:** 000000B8h at 20h **Offset End:** 000000BBh at 20h
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0		**Offset Start:** 000000B8h at 20h **Offset End:** 000000BBh at 20h
Size: 32 bit	**Default:** XXXXXXXXh				**Power Well:** Core
Bit Range	**Bit Acronym**	**Bit Description**		**Bit Reset Value**	**Bit Access**
31 :16	RSVD	Reserved		Xh	RW
15 :00	Data_63_48	Data (bits [63:48])		Xh	RW

39.6.1.18 Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data

Note: These registers are implemented in the SRAM which does not have the capability to mask writes to reserved bits. Therefore, reserved bits in this CSR will be RW. Software should treat these bits as reserved and not change the reset value of these bits.

Note: These registers are implemented in SRAM which is not initialized at power-up or upon reset. So before enabling the CAN, software needs to update these CSR's with the reset values.

Table 39-23. Offset 000000BCh: RxMessageACR_Data[0-15] - Receive Message ACR Data

Description:					
View: PCI 1	**BAR:** CSRBAR		**Bus:Device:Function:** M:4:0	**Offset Start:** 000000BCh at 20h **Offset End:** 000000BFh at 20h	
View: PCI 2	**BAR:** CSRBAR		**Bus:Device:Function:** M:5:0	**Offset Start:** 000000BCh at 20h **Offset End:** 000000BFh at 20h	
Size: 32 bit	**Default:** XXXXXXXXh			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	RSVD	Reserved.		Xh	RW
15 :00	Data_63_48	Data (bits [63:48])		Xh	RW

§ §

40.0 SSP Serial Port

40.1 Overview

The SSP (Synchronous Serial Port) is a full-duplex synchronous serial interface. The SSP can connect to a variety of external analog-to-digital (A/D) converters, audio and telecom codecs, and many other devices that use serial protocols for transferring data. It supports National Microwire*, Texas Instruments* synchronous serial protocol (SSP), and Motorola* serial peripheral interface (SPI) protocol.

The SSP operates in master mode (the attached peripheral functions as a slave), and supports serial bit rates from 7.2 Kbps to 1.84 Mbps. Serial data formats may range from 4 to 16 bits in length. Two on-chip register blocks function as independent FIFOs for data, one for each direction. The buffers are 16 entries deep x 16 bits wide.

This section describes the signal definitions and operations of the SSP functional block.

40.2 Feature List

A list of features is presented below:

- Supports National Microwire format.
- Supports Texas Instruments Synchronous Serial Protocol (SSP).
- Supports Motorola Serial Peripheral Interface (SPI).
- Supports serial data rates from 7.2 Kbps to 1.84 Mbps.
- Provides 32 byte FIFOs for both receive and transmit data.

40.3 Theory of Operation

40.3.1 Endianness

The SSP uses only bits 15:0 of the internal data bus, the host must format the data into the two least significant bytes of the 32 bit internal bus transfer. The two high order bytes are ignored. The SSP unit is accessed only using dword accesses.

40.3.2 Error Handling

No error handling is defined for this unit beyond the functional receiver overrun status bit (ROR) (see Section 40.4.3.7) in the SSP Status Register (see Section 40.4.3). This error condition causes the unit's interrupt to be asserted and is non-maskable.

40.4 Register Summary

There are five registers in the SSP block: two control, one data, one status register, and one test register.

- Control registers are used to program the baud rate, data length, frame format, data transfer mechanism, and port enabling. In addition, they permit setting the FIFO “fullness” threshold that will trigger an interrupt.
- The Data Register is mapped as one 32-bit location, which physically points to either of two 32-bit registers. One register is for WRITES, and transfers data to the Transmit FIFO; the other is for READS, and takes data from the Receive FIFO. A write cycle, will load successive words into the SSP Write Register, from the lower half 2 bytes of a 32-bit word to the Transmit FIFO. A READ cycle, will similarly take data from the SSP Read Register, and the Receive FIFO will reload it with available data bits it has stored.

  The FIFOs are independent buffers that allow full duplex operation.
- The Status Register signals the state of the FIFO buffers: whether the programmable threshold has been passed (Transmit/Receive Buffer service request), and a value showing the actual “fullness” of the FIFO. There are flag bits to indicate when the SSP is actively transmitting data, when the Transmit Buffer is not full, and when the Receive Buffer is not empty. Error bits signal overrun errors.

The SSP registers materialize in the PCI space.

Table 40-1 summarizes the SSP materialization from the PCI perspective.

Table 40-1. Bus M, Device 6, Function 0: Summary of SSP CSRs

Offset Start	Offset End	Register ID - Description	Default Value
00h	03h	“Offset 00h: SSCR0 - SSP Control Register 0 Details” on page 1607	00000000h
04h	07h	“Offset 04h: SSCR1 - SSP Control Register 1 Details” on page 1610	00000000h
08h	0Bh	“Offset 08h: SSSR - SSP Status Register Details” on page 1614	0000F004h
0Ch	0Fh	“Offset 0Ch: SSITR - SSP Interrupt Test Register Details” on page 1617	00000000
10h	13h	“Offset 10h: SSDR - SSP Data Register Details” on page 1618	00000000h

40.4.1 SSP Control Register 0

40.4.1.1 Offset 00h: SSCR0 - SSP Control Register 0 Details

The SSP control register 0 (SSCR0) contains five different bit fields that control various functions within the SSP.

Register Name:	SSCR0																													
Block Base Address:	N/A						Offset Address						00						Reset Value						00000000					
Register Description:	SSC Control Register 0																						Access:		(See below.)					
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																SCR								SSE	ECS	FRF		SRS			

Table 40-2. Offset 00h: SSCR0 - SSP Control Register 0 Details (Sheet 1 of 2)

Description:	SSP Control Register 0				
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:6:0	Offset Start: 00h Offset End: 03h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		0h	RV
15 :08	SCR	Serial Clock Rate Selection Value (0 to 255) used to generate transmission rate of SSP. Bit rate = CLK/ (2 x (SCR + 1)) where SCR is a decimal integer CLK may be the internally provided clock of 3.7 MHz (2.777MHz for low-power SKU) or the externally provided clock.		0h	RW
7	SSE	Synchronous Serial Port Enable bit. 0 = SSP operation disabled 1 = SSP operation enabled		0b	RW

Table 40-2. Offset 00h: SSCR0 - SSP Control Register 0 Details (Sheet 2 of 2)

Description:	SSP Control Register 0				
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:6:0	**Offset Start:** **00**h **Offset End:** 03h	
Size: 32 bit	**Default:** **00000000**h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
6	ECS	External clock select bit. 0 = On-chip clock used to produce the SSP's serial clock (**SSP_SCLK**). 1 = **SSP_EXTCLK** is used to create the SSP's **SSP_SCLK**.		0b	RW
05 :04	FRF	This field specifies the FRame Format. 00 - Motorola* Serial Peripheral Interface (SPI) 01 - Texas Instruments* Synchronous Serial Protocol (SSP) 10 - National Microwire* 11 - Reserved, undefined operation		0h	RW
03 :00	DSS	This field specifies the Data Size Selection. 0000 - Reserved, undefined operation 0001 - Reserved, undefined operation 0010 - Reserved, undefined operation 0011 - 4-bit data 0100 - 5-bit data 0101 - 6-bit data 0110 - 7-bit data 0111 - 8-bit data 1000 - 9-bit data 1001 - 10-bit data 1010 - 11-bit data 1011 - 12-bit data 1100 - 13-bit data 1101 - 14-bit data 1110 - 15-bit data 1111 - 16-bit data		0h	RW

40.4.1.2 Data Size Select (DSS)

The 4-bit data size select (DSS) field is used to select the size of the data transmitted and received by the SSP. Data can be 4 to 16 bits in length. When data is programmed to be less than 16 bits, received data is automatically right-justified and the upper bits in the receive FIFO are zero-filled by receive logic. Transmit data should not be left-justified by the user before being placed in the transmit FIFO; transmit logic in the SSP will automatically left-justify the data sample according to the value of DSS before the sample is transmitted on **SSP_TXD**. Although it is possible to program data sizes of 1, 2, and 3 bits, these sizes are reserved and produce unpredictable results in the SSP.

When National Microwire frame format is selected, this bit field selects the size of the received data. Note that the size of the transmitted data is always 8 bits in this mode.

40.4.1.3 Frame Format (FRF)

The 2-bit frame format (FRF) field is used to select which frame format to use: Motorola SPI (FRF=00), Texas Instruments synchronous serial (FRF=01), or National Microwire (FRF=10). Note that FRF=11 is reserved and the SSP will produce unpredictable results if this value is used.

40.4.1.4 External Clock Select (ECS)

The external clock select (ECS) bit selects whether the on-chip 3.6864-MHz (2.777MHz for low-power SKU) clock is used by the SSP or if an off-chip clock is supplied via **SSP_EXTCLK**. When ECS=0, the SSP uses the on-chip 3.6864-MHz clock (2.777MHz for low-power SKU) to produce a range of serial transmission rates. When ECS=1, the SSP uses **SSP_EXTCLK** to input a clock supplied from off-chip. The frequency of the off-chip clock can be any value up to 3.6864 MHz (2.777MHz for low-power SKU). This off-chip clock is useful when a serial transmission rate, which is not an even multiple of the internal clock, is required for synchronization with the target off-chip slave device.

40.4.1.5 Synchronous Serial Port Enable (SSE)

The SSP enable (SSE) bit is used to enable and disable all SSP operations. When SSE=0, the SSP is disabled; when SSE=1, it is enabled. When the SSP is disabled, all of its clocks are powered down to minimize power consumption. Note that the SSE is the only control bit within the SSP that is reset to a known state. It is cleared to zero to ensure the SSP is disabled following a reset.

When the SSE bit is cleared during active operation, the SSP is disabled immediately, causing the current frame being transmitted to be terminated. Clearing SSE resets the SSP's FIFOs. However the SSP's control and status registers are not reset. The user must ensure these registers are properly reconfigured before re-enabling the SSP.

40.4.1.6 Serial Clock Rate (SCR)

The 8-bit serial clock rate (SCR) bit-field is used to select the baud, or bit rate, of the SSP. The serial clock generator can be configured to use the internally provided 3.6864-MHz (2.777MHz for low-power SKU) clock produced by the on-chip PLL or the externally provided clock. The source clock is divided by a fixed value of 2, and then divided by the programmable SCR value (0 to 255) plus 1 to generate the serial clock (**SSP_SCLK**). When configured to use the internal clock, a total of 256 different bit rates can be selected. The resultant clock is driven on the **SSP_SCLK** pin and is used by the SSP's transmit logic to drive data on the **SSP_TXD** pin, and to latch data on the **SSP_RXD** pin. Depending on the frame format selected, each transmitted bit is driven on either the rising or falling edge of **SSP_SCLK**, and is sampled on the opposite clock edge.

Note that the SSE bit is the only control bit that is reset to a known state, to ensure the SSP is disabled following a reset. The reset state of all other control bits is unknown and must be initialized before enabling the SSP.

40.4.2 SSP Control Register 1

40.4.2.1 Offset 04h: SSCR1 - SSP Control Register 1 Details

The SSP Control Register 1 (SSCR1) contains nine bit fields that control various SSP functions.

Register Name:	SSCR1																													
Block Base Address:	N/A								Offset Address	0x04										Reset Value	00000000									
Register Description:	SSC Control Register 1																			Access:	(See below.)									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																STRF	EFWR	RFT				TFT				MWDS	SPH	SPO	LBM	TIE	RIE

Table 40-3. Offset 04h: SSCR1 - SSP Control Register 1 Details (Sheet 1 of 2)

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:6:0	Offset Start: 04h Offset End: 07h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		0h	RV
15	STRF	Select FIFO for EFWR: 0 = Transmit FIFO is selected for “enable FIFO write/ read” 1 = Receive FIFO is selected for “enable FIFO write/read”		0b	RW
14	EFWR	Enable FIFO Write/Read: 0 = FIFO write/read function is disabled 1 = FIFO write/read function is enabled		0b	RW
13 :10	RFT	Receive FIFO Threshold. Sets threshold level at which Receive FIFO asserts interrupt. This level should be set to the threshold value minus 1.		0h	RW
09 :06	TFT	Transmit FIFO Threshold. Sets threshold level at which Transmit FIFO asserts interrupt. This level should be set to the threshold value minus 1.		0h	RW
05	MWDS	National Microwire* Data Size 0 = 8 bit Microwire format 1 = 16 bits Microwire format		0b	RW
04	SPH	Motorola* SPI **SSP_SCLK** phase setting: 0 = **SSP_SCLK** is inactive one full cycle at the start of a frame and 1/2 cycle at the end of a frame. 1 = **SSP_SCLK** is inactive 1/2 cycle at the start of a frame and one full cycle at the end of a frame.		0b	RW

Table 40-3. Offset 04h: SSCR1 - SSP Control Register 1 Details (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:6:0	**Offset Start:** **04**h **Offset End:** 07h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
03	SPO	Motorola SPI **SSP_SCLK** polarity setting: 0 = The inactive or idle state of **SSP_SCLK** is low. 1 = The inactive or idle state of **SSP_SCLK** is high.		0b	RW
02	LBM	Loop Bank Mode Enable bit. 0 = Normal serial port operation enabled 1 = Output of transmit serial shifter connected to input of receive serial shifter, internally		0b	RW
01	TIE	Transmit FIFO Interrupt Enable 0 = Transmit FIFO level interrupt is disabled 1 = Transmit FIFO level interrupt is enabled		0b	RW
00	RIE	Receive FIFO Interrupt Enable 0 = Receive FIFO level interrupt is disabled 1 = Receive FIFO level interrupt is enabled		0b	RW

40.4.2.2 Receive FIFO Interrupt Enable (RIE)

The Receive FIFO Interrupt Enable (RIE) bit is used to mask or enable the Receive FIFO service request interrupt. When RIE=0, the interrupt is masked and the state of the Receive FIFO Service Request (RFS) bit within the SSP Status Register is ignored by the interrupt controller. When RIE=1, the interrupt is enabled and whenever RFS is set to one, an interrupt request is made to the interrupt controller. Note that programming RIE=0 does not affect the current state of RFS or the receive FIFO logic's ability to set and clear RFS, it only blocks the generation of the interrupt request.

40.4.2.3 Transmit FIFO Interrupt Enable (TIE)

The Transmit FIFO Interrupt Enable (TIE) bit is used to mask or enable the transmit FIFO service request interrupt. When TIE=0, the interrupt is masked and the state of the Transmit FIFO Service Request (TFS) bit within the SSP Status Register is ignored by the interrupt controller. When TIE=1, the interrupt is enabled, and whenever TFS is set to one an interrupt request is made to the interrupt controller. Note that programming TIE=0 does not affect the current state of TFS or the transmit FIFO logic's ability to set and clear TFS, it only blocks the generation of the interrupt request.

40.4.2.4 Loop Back Mode (LBM)

The loop back mode (LBM) bit is used to enable and disable the ability of the SSP transmit and receive logic to communicate. When LBM=0, the SSP operates normally. The transmit and receive data paths are independent and communicate via their respective pins. When LBM=1, the output of the transmit serial shifter is directly connected to the input of the receive serial shifter internally.

Loop back mode is only valid for SSP and SPI modes, Microwire mode does not support loop back mode testing because the bus protocol is half-duplex.

Note: While in loopback mode, the data will continue to be driven on the transmit pins.

40.4.2.5 Serial Clock Polarity (SPO)

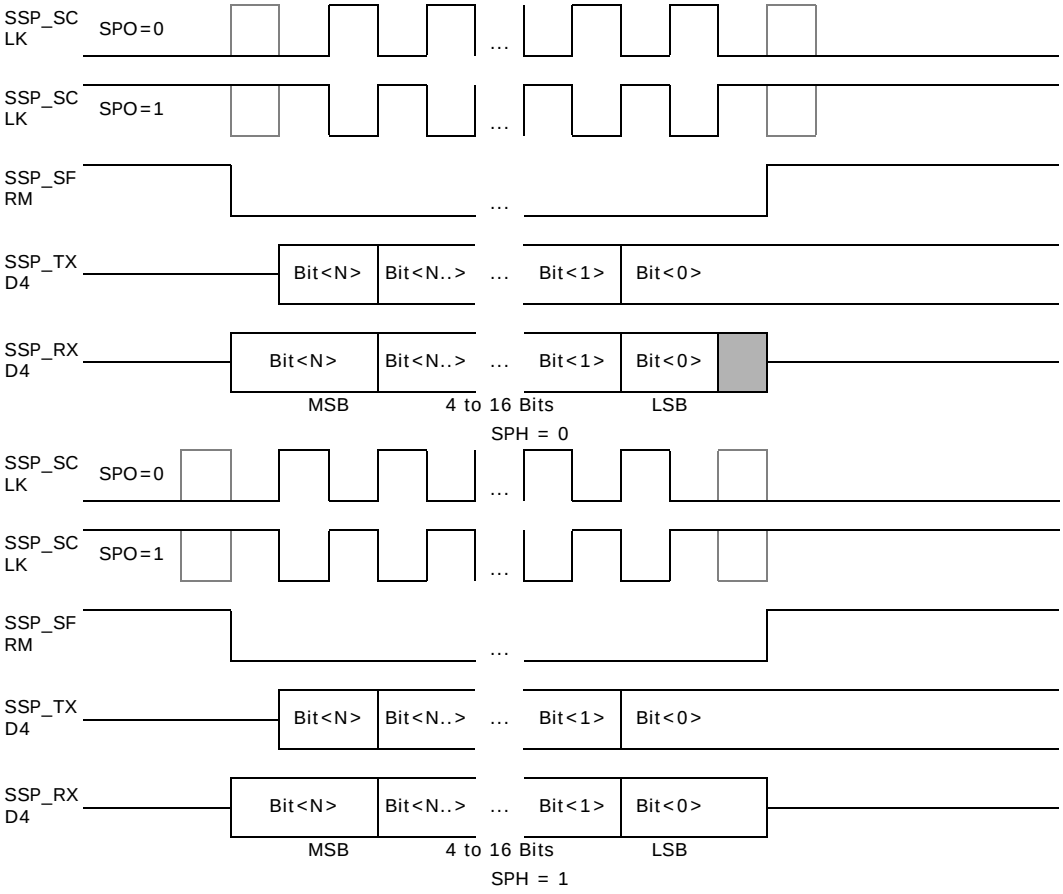
The serial clock (**SSP_SCLK**) polarity bit (SPO) selects the polarity of the inactive state of the **SSP_SCLK** pin when Motorola SPI format is selected (FRF=00). For SPO=0, the **SSP_SCLK** is held low in the inactive or idle state when the SSP is not transmitting/ receiving data. For SPO=1, the **SSP_SCLK** is held high during the inactive/idle state. The programmed setting of the SPO alone does not determine which **SSP_SCLK** edge is used to transmit or receive data. The SPO setting in combination with the **SSP_SCLK** phase bit (SPH) determines this. Note that the SPO is ignored for all data frame formats except for the Motorola SPI format (FRF=00).

40.4.2.6 Serial Clock Phase (SPH)

The serial clock (**SSP_SCLK**) phase bit (SPH) determines the phase relationship between the **SSP_SCLK** and the serial frame (**SSP_SFRM**) pins when the Motorola SPI format is selected (FRF=00). When SPH=0, **SSP_SCLK** remains in its inactive/idle state (as determined by the SPO setting) for one full cycle after **SSP_SFRM** is asserted low at the beginning of a frame. **SSP_SCLK** continues to transition for the rest of the frame and is then held in its inactive state for one-half of an **SSP_SCLK** period before **SSP_SFRM** is de-asserted high at the end of the frame. When SPH=1, **SSP_SCLK** remains in its inactive/idle state (as determined by the SPO setting) for one-half cycle after **SSP_SFRM** is asserted low at the beginning of a frame. **SSP_SCLK** continues to transition for the rest of the frame and is then held in its inactive state for one full **SSP_SCLK** period before **SSP_SFRM** is de-asserted high at the end of the frame. The combination of the SPO and SPH settings determines when **SSP_SCLK** is active during the assertion of **SSP_SFRM** and which **SSP_SCLK** edge is used to transmit and receive data on the **SSP_TXD** and **SSP_RXD** pins. When SPO and SPH are programmed to the same value (both 0 or both 1), transmit data is driven on the falling edge of **SSP_SCLK** and receive data is latched on the rising edge of **SSP_SCLK**. When SPO and SPH are programmed to opposite values (one 0 and the other 1), transmit date is driven on the rising edge of **SSP_SCLK** and receive data is latched on the falling edge of **SSP_SCLK**. Note that the SPH is ignored for all data frame formats except for the Motorola SPI format (FRF=00).

Table 40-4 shows the pin timing for all four programming combinations of SPO and SPH. Note that SPO inverts the polarity of the **SSP_SCLK** signal and SPH determines the phase relationship between **SSP_SCLK** and **SSP_SFRM**, shifting the **SSP_SCLK** signal one-half phase to the left or right during the assertion of **SSP_SFRM**.

Table 40-4. Motorola* SPI Frame Formats for SPO and SPH Programming



40.4.2.7 National Microwire* Data Size (MWDS)

This bit sets the size of data in the Microwire format. If '1', a 16 bits data size is chosen for the Microwire format, otherwise, an 8-bit data size.

40.4.2.8 Transmit FIFO Interrupt Threshold (TFT)

This 4-bit value sets the level at or below which the FIFO controller triggers a service interrupt.

40.4.2.9 Receive FIFO Interrupt Threshold (RFT)

This 4-bit value sets the level at or above which the FIFO controller triggers a service interrupt.

40.4.2.10 Enable FIFO Write/Read Function (EFWR)

This bit enables a special functional mode for the SSP. When EFWR = 0, the SSP operates in the normal mode described in this document. When EFWR = 1, the SSP enters a mode in which whenever the CPU reads or writes to the SSP Data register, it actually reads and writes exclusively to either the Transmit FIFO or the Receive FIFO depending on the programmed state of the Select FIFO for EFWR (STRF) bit. In this special mode, data will not be transmitted on the TXD pin and data input on the RXD

pin will not be stored. This mode can be used to test, through software, whether or not the Transmit FIFO or the Receive FIFO operates properly as a first-in-first-out memory stack.

Note: When this mode is enabled, a write followed immediately by a read to the RX or TX FIFOs may not return the correct read data. Since this is a test mode, a delay should be inserted between the write and read transactions so that the data is guaranteed to be read back correctly.

40.4.2.11 Select FIFO for Enable FIFO Write/Read (STRF)

This bit selects whether the Transmit or Receive FIFO is enabled for write/read.

40.4.3 SSP Status Register

40.4.3.1 Offset 08h: SSSR - SSP Status Register Details

The SSP status register (SSSR) contains bits that signal overrun errors as well as the transmit and receive FIFO service requests. Each of these hardware-detected events signal an interrupt request to the interrupt controller. The status register also contains flags that indicate when the SSP is actively transmitting characters, when the transmit FIFO is not full, and when the receive FIFO is not empty (no interrupt generated).

Bits that cause an interrupt will signal the request as long as the bit is set. Once the bit is cleared, the interrupt is cleared. Read/write bits are called status bits, read-only bits are called flags. Status bits are referred to as “sticky” (once set by hardware, must be cleared by software). Writing a 1 to a sticky status bit clears it, writing a 0 has no effect. Read-only flags are set and cleared by hardware; writes have no effect. Additionally some bits that cause interrupts have corresponding mask bits in the control registers and are indicated in the section headings that follow.

Register Name:	SSSR																													
Block Base Address:	N/A							Offset Address	08h										Reset Value	0000F024									
Register Description:	SSP Status Register																		Access:	(See below.)									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																RFL				TFL				ROR	RFS	TFS	BSY	RNE	TNF	Rsvd.	

Table 40-5. Offset 08h: SSSR - SSP Status Register Details (Sheet 1 of 2)

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:6:0	Offset Start: 08h Offset End: 0Bh	
Size: 32 bit	Default: 0000F004h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		0h	RV
15 :12	RFL	Receive FIFO Level: Number of entries in Receive FIFO		Fh	RO
11 :08	TFL	Transmit FIFO Level: Number of entries in Transmit FIFO		0h	RO

Table 40-5. Offset 08h: SSSR - SSP Status Register Details (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:6:0	**Offset Start:** **08**h **Offset End:** 0Bh	
Size: 32 bit	**Default:** 0000F004h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
07	ROR	Receive FIFO Overrun: 0 = Receive FIFO has not experienced an overrun 1 = Attempted data write to full Receive FIFO, request interrupt	y	0b	RWC
06	RFS	Receive FIFO Service Request: 0 = Receive FIFO level is below RFT threshold, or SSP disabled. 1 = Receive FIFO level is at or above RFL threshold, request interrupt		0b	RO
05	TFS	Transmit FIFO Service Request: 0 - Transmit FIFO level exceeds TFT threshold, or SSP disabled 1 - Transmit FIFO level is at or below TFL threshold, request interrupt		0b	RO
04	BSY	SSP is busy 0 = SSP is idle or disabled 1 = SSP currently transmitting or receiving a frame		0b	RO
03	RNE	Receive FIFO not empty. 0 - Receive FIFO is empty 1 - Receive FIFO is not empty		0b	RO
02	TNF	Transmit FIFO not Full. 0 = Transmit FIFO is full 1 = Transmit FIFO is not full		1b	RO
01 :00	Reserved	Reserved		0h	RV

40.4.3.2 Transmit FIFO Not Full Flag (TNF) (Read-Only, Non-Interruptible)

The transmit FIFO not full flag (TNF) is a read-only bit that is set whenever the transmit FIFO contains one or more entries that do not contain valid data. TNF is cleared when the FIFO is completely full. This bit can be polled when using programmed I/O to fill the transmit FIFO over its half-way mark. This bit does not request an interrupt.

40.4.3.3 Receive FIFO Not Empty Flag (RNE) (Read-Only, Non-Interruptible)

The receive FIFO not empty flag (RNE) is a read-only bit that is set whenever the receive FIFO contains one or more entries of valid data and is cleared when it no longer contains any valid data. This bit can be polled when using programmed I/O to remove remaining bytes of data from the receive FIFO since CPU interrupt requests are only made when the Receive FIFO Threshold has been met or exceeded. This bit does not request an interrupt.

40.4.3.4 SSP Busy Flag (BSY) (Read-Only, Non-Interruptible)

The SSP busy (BSY) flag is a read-only bit that is set when the SSP is actively transmitting and/or receiving data and is cleared when the SSP is idle or disabled (SSE=0). This bit does not request an interrupt.

40.4.3.5 Transmit FIFO Service Request Flag (TFS) (Read-Only, Maskable Interrupt)

The Transmit FIFO service request flag (TFS) is a read-only bit that is set when the transmit FIFO is nearly empty and requires service to prevent an underrun. TFS is set any time the transmit FIFO has the same or fewer entries of valid data than indicated by the Transmit FIFO Threshold and it is cleared when it has more entries of valid data than the threshold value. When the TFS bit is set, an interrupt request is made unless the transmit FIFO interrupt request enable (TIE) bit is cleared. After the CPU fills the FIFO such that it exceeds the threshold, the TFS flag (and the service request and/or interrupt) is automatically cleared.

40.4.3.6 Receive FIFO Service Request Flag (RFS) (Read-Only, Maskable Interrupt)

The receive FIFO service request flag (RFS) is a read-only bit that is set when the receive FIFO is nearly filled and requires service to prevent an overrun. RFS is set any time the receive FIFO has the same or more entries of valid data than indicated by the Receive FIFO Threshold and it is cleared when it has fewer entries than the threshold value. When the RFS bit is set, an interrupt request is made unless the receive FIFO interrupt request enable (RIE) bit is cleared. After the CPU reads the FIFO such that it has fewer entries than the RFT value, the RFS flag (and the service request and/or interrupt) is automatically cleared.

40.4.3.7 Receiver Overrun Status (ROR) (Read/Write, Non-Maskable Interrupt)

The receiver overrun status bit (ROR) is a read/write bit that is set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. Each time a new piece of data is received, the set signal to the ROR bit is asserted and the newly received data is discarded. This process is repeated for each new piece of data received until at least one empty FIFO entry exists. When the ROR bit is set, an interrupt request is made. Writing 1 to this bit resets ROR status and its interrupt request.

40.4.3.8 Transmit FIFO Level

This 4-bit value shows how many valid entries are currently in the Transmit FIFO.

40.4.3.9 Receive FIFO Level

This 4-bit value shows how many valid entries are currently in the Receive FIFO.

The following bit table shows the bit locations corresponding to the status and flag bits within the SSP status register. All bits are read-only except ROR, which is read/write. Writes to TNF, RNE, BSY, TFS, and RFS have no effect. The reset state of ROR is unknown and must be initialized before enabling the SSP. Note that writes to reserved bits are ignored and reads to those bits return zeros.

40.4.4 SSP Interrupt Test Register

40.4.4.1 Offset 0Ch: SSITR - SSP Interrupt Test Register Details

Writing ‘1’ to the corresponding bit position to the SSP Interrupt Test register generates an interrupt strobe signal to the Interrupt Controller for test purposes.

Note: SSITR functionality is available even when the SSP is disabled.

Register Name:	SSITR																															
Block Base Address:	N/A			Offset Address	0x0C														Reset Value	00000000											
Register Description:	SSP Interrupt Test Register																						Access:	(See below.)							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								TROR	TRFS	TTFS	Reserved				

Table 40-6. Offset 0Ch: SSITR - SSP Interrupt Test Register Details

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:6:0	Offset Start: 0Ch Offset End: 0Fh	
Size: 32 bit	Default: 00000000			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :08	Reserved	Reserved		0h	RV
07	TROR	Test Receive FIFO overrun (ROR)		0b	RW
06	TRFS	Test Receive FIFO service request (RFS)		0b	RW
05	TTFS	Test Transmit FIFO service request (TFS)		0b	RW
04 :00	Reserved	Reserved		0h	RV

40.4.5 SSP Data Register

40.4.5.1 Offset 10h: SSDR - SSP Data Register Details

The SSP Data Register (SSDR) is a block of 32-bit locations that can be accessed by 32-bit data transfers. Transfers can be from 1 to 8 words. The SSDR represents two physical registers: the first is temporary storage for data on its way out through the Transmit FIFO, the other is temporary storage for data coming in through the Receive FIFO.

As the register is accessed by the system, FIFO control logic transfers data automatically between register and FIFO as fast as the system moves it. Data in the FIFO shifts up or down to accommodate the new word (unless it's an attempted WRITE to a full Transmit FIFO). Status bits are available to show the system whether either buffer is full, above/below a programmable threshold, or empty.

For outbound data transfers (WRITE from system to SSP peripheral), the register may be loaded (written) by the system processor anytime the register is empty.

When a data size of less than 16-bits is selected, the user should not left-justify data written to the transmit FIFO. Transmit logic left-justifies the data and ignores any unused bits. Received data less than 16-bits is automatically right-justified in the receive buffer. When the SSP is programmed for National Microwire frame format, the default size for transmit data is 8-bits (the most significant byte is ignored), the receive data size is controlled by the programmer using the DSS field in SSCR0.

The following table shows the location of the SSP data register. Note that both FIFOs are cleared when the block is reset or by writing a zero to SSE (SSP disabled).

Register Name:	SSDR						
Block Base Address:	N/A		Offset Address	0x10		Reset Value	00000000
Register Description:	SSP Data Register					Access:	(See below.)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Data															

Table 40-7. Offset 10h: SSDR - SSP Data Register Details

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:6:0	Offset Start: 10h Offset End: 13h	
Size: 32 bit	Default: 00000000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :16	Reserved	Reserved		0h	RV
15 :00	Data	Data (Low Word): When written, the data will be written to the Transmit FIFO. When read, data from the Receive FIFO is returned.		0h	RW

§ §

41.0 IEEE 1588 Time Synchronization Hardware Assist

41.1 Overview

This document describes the hardware-assist logic developed to achieve time synchronization on Ethernet and CAN.

In a distributed control system containing multiple clocks, individual clocks tend to drift apart. Some kind of correction mechanism is necessary to synchronize the individual clocks to maintain global time, which is accurate to some requisite clock resolution. For this purpose, you can use the IEEE 1588 standard, which defines a precision clock synchronization protocol for networked measurement and control systems. The IEEE standard defines several messages that you can use to exchange timing information. The hardware assist logic, required to achieve precision clock synchronization using the IEEE 1588 standard, is left to the implementation.

The time synchronization logic monitors the internal MII and GMII signals on the Gigabit Ethernet controller. Only Gigabit port 0 and port 1 are supported. The interrupt signals from CAN will also be monitored to allow CAN traffic to be time-stamped as well.

An interrupt signal to the host processor is generated by the 1588 Time Sync block when any of the following conditions occur and are enabled:

- Target Time expiration
- Auxiliary Target Time expiration
- Auxiliary Master Mode snapshot is taken
- Auxiliary Slave Mode snapshot is taken
- Pulse Per Second signal assertion

The System Time, Target Time and Aux Target Time registers are all 64 bits wide. When the system time is greater than or equal to the target time, the target time expiration condition will set. An interrupt enable mask must be set to allow the target time interrupt to pass to the core.The Aux Target Time register works the same as the Target Time register.

41.2 Feature List

A list of features is presented below:

- Supports IEEE1588-2008.
- Supports user-defined messages.
- Supports detection of IEEE 1588 messages for both receive and transmit directions.
- Supports Ethernet MII and GMII.
- Supports CAN 2.0B.
- Provides external signals to signify if a snapshot was taken.
- Provides external control over taking snapshots.

- Supports 15 ns resolution on the 1066 MHz and 1200 MHz SKUs.
- Supports 20 ns resolution on the 600MHz SKUs.
- Provides a Pulse Per Second output.

41.2.1 Signal Descriptions

- **Auxiliary Slave Mode Snapshot** – An active high level on this input causes a snapshot of system time to be captured in the ASMS register. Refer to "Auxiliary Snapshots" on page 4765 for usage and restrictions.
- **Auxiliary Master Mode Snapshot** – An active high level on this input causes a snapshot of system time to be captured in the AMMS register. Refer to "Auxiliary Snapshots" on page 4765 for usage and restrictions.
- **Test Mode Data –** This signal will reflect bits of the internal System Timer depending on the setting of internal control bits in the TS_Test register.
- **Pulse Per Second Output** - This signal is asserted high when a match occurs between the Compare register and lower 32 bits of system time. Clearing of this signal is under firmware control. The register and pin can be used to create a pulse per second event.
- **Snapshot Taken Outputs –** These 2 signals will each pulse high for eight (8) pclks whenever a timestamp has been taken on the selected channel. (either TX or RX). The channel that is monitored is determined by the control bits in the TS_Test Register.

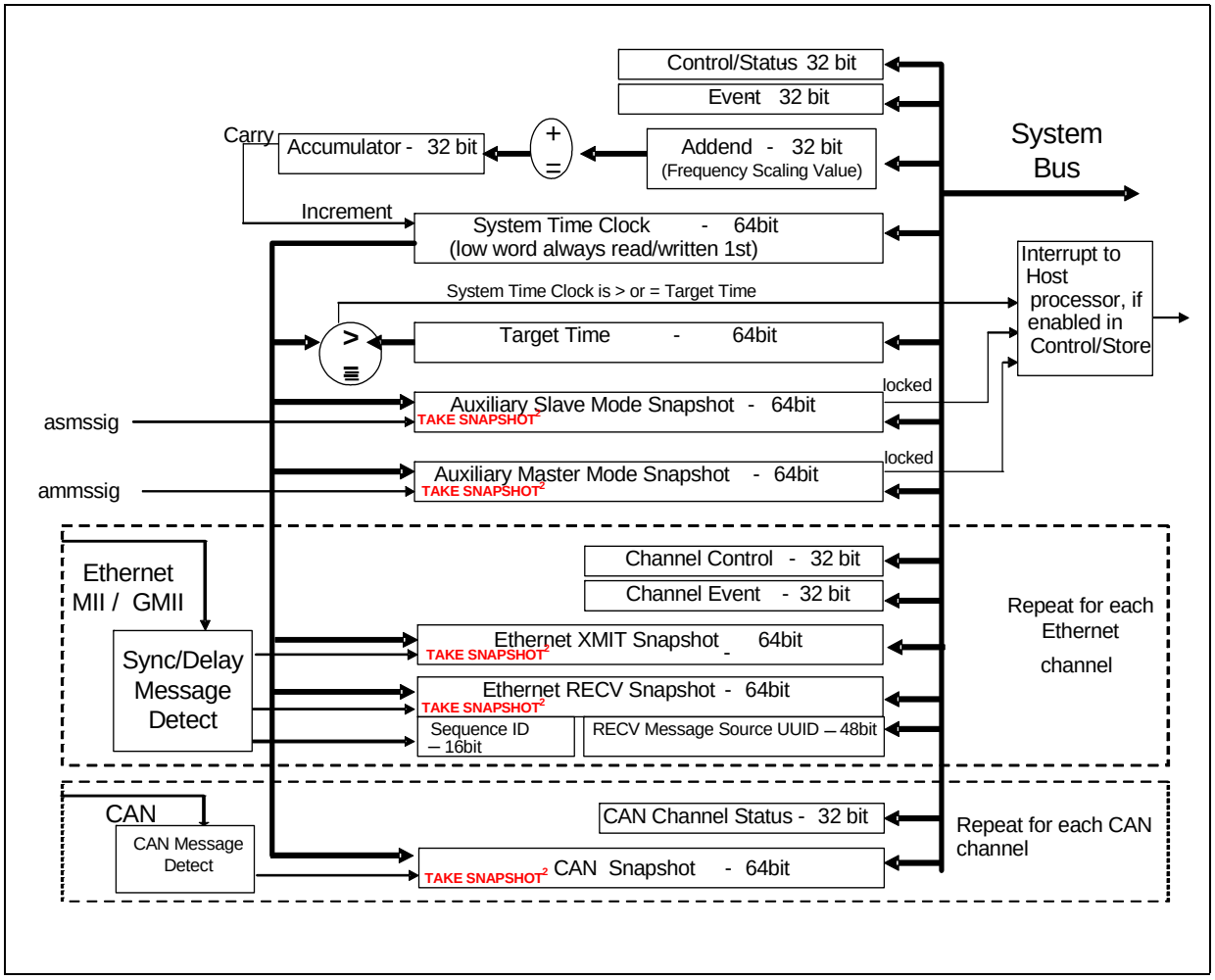
41.3 Functional Block Diagram

A programming model is shown in Figure 41-1, showing registers and interconnections.

Note: If the functional block is the Master, the XMIT snapshot holds the "SYNC" message time and the RECV snapshot holds the DELAY" message time. If the unit is the Slave, the XMIT snapshot holds the "DELAY" message time and the RECV snapshot holds the "SYNC" message time.

Note: "Take snapshot" = Synch, edge detect, and lock, until reset by writing a "1" to the corresponding bit in the event register.

Figure 41-1. Programming Model



41.4 Usage Model

41.4.1 Channel Mapping

Ten channels are supported which are mapped to the Ethernet and CAN interfaces as shown in the table below:

Table 41-1. Channel Mapping to Interfaces

Channel	Interface Being Monitored	Notes
Ethernet Channel 0	N/A	Reserved for future use
Ethernet Channel 1	N/A	Reserved for future use
Ethernet Channel 2	N/A	Reserved for future use
Ethernet Channel 3	Gbe 0 GMII/MII	Both GMII and MII modes supported
Ethernet Channel 4	Gbe 1 GMII/MII	Both GMII and MII modes supported
Ethernet Channel 5	N/A	Reserved for future use
Ethernet Channel 6	N/A	Reserved for future use
Ethernet Channel 7	N/A	Reserved for future use
CAN Channel 0	CAN 0	Supports simple timestamping of packets
CAN Channel 1	CAN 1	Supports simple timestamping of packets

41.5 Functional Description

41.5.1 IEEE 1588 Overview

A simplified view of an example 1588 Network topology is shown in Figure 41-2, “Example Network Topology” on page 1623. Each device in an IEEE 1588 enabled network may have one or more ports each of which is defined to be a master or a slave port. Each slave synchronizes it’s internal clock to that of the master. Each master-enabled port may also act as a slave. The specific hierarchy of the network is determined dynamically.

There are five basic types of Precision Time Protocol (PTP) devices:

- Ordinary clock.
- Boundary clock.
- End-to-end transparent clock.
- Peer-to-peer transparent clock.
- Management node.

An **ordinary clock** communicates with the network via a single physical port. This interface is used to send and receive PTP messages, which are time stamped based on the value of the local clock. The ordinary clock can be the grandmaster clock in a system or it can be a slave clock in the master-slave hierarchy.

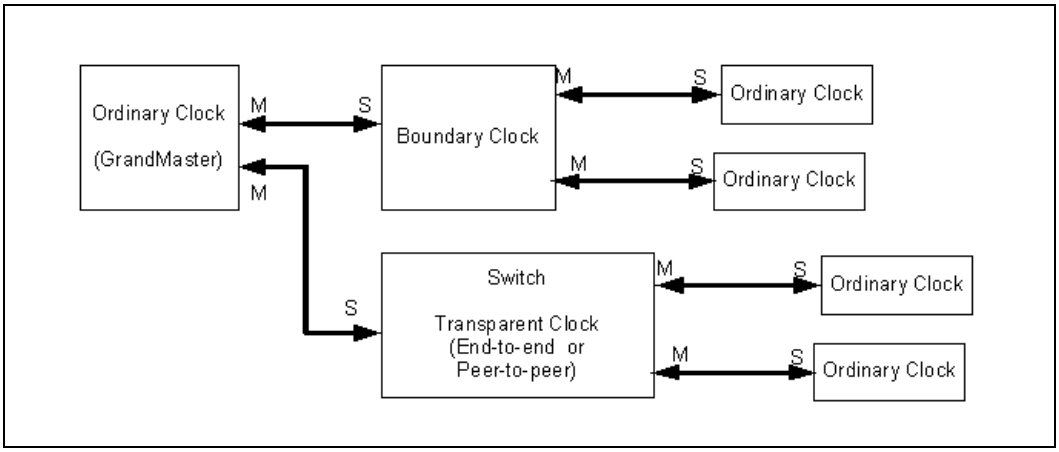
The **boundary clock** typically has several physical ports. Each port of a boundary clock is similar to the port of an ordinary clock, but the local clock is now common to all the ports. Each port of the boundary clock terminates all messages related to synchronization and establishing the master-slave hierarchy.

The **end-to-end transparent clock** forwards all messages just as a normal switch, router or repeater. However for PTP-event messages, the time the message takes to traverse the switch is measured and appended to the forwarded event messages.

The **peer-to-peer transparent clock** is similar to the end-to-end transparent clock however, it also measures peer-to-peer link delays in addition to the internal message traversal time.

A **management node** is a PTP device that is physically connected to the network and provides an user interface to the PTP management messages. It will not be discussed further here

Figure 41-2. Example Network Topology



The operation of a PTP enabled network is divided into two stages, Initialization and time synchronization.

41.5.1.1 Initialization

At the initialization stage every master enabled node starts by sending Sync packets that include the clock parameters of its clock. These parameters include the source of the clock, the accuracy of the clock, it's variance, etc. Upon reception of a Sync packet, a node compares the received clock parameters to its own and if the received parameters are better, then this node moves to Slave state and stops sending Sync packets. When in slave state the node continuously compares the incoming packet to its currently chosen master and if the new clock parameters are better then the master selection is transferred to this master clock. Eventually the best master clock is chosen. Every node has a defined time-out interval in which if no Sync packet was received from its chosen master clock it moves back to master state and starts sending Sync packets until a new best master clock (BMC) is chosen.

41.5.1.2 Time Synchronization

The time synchronization stage is different to master and slave nodes. If a node is at master state it should periodically send a Sync packet which is time stamped by hardware on the TX path (as close as possible to the PHY). After the Sync packet is sent, a Follow_Up packet is sent which includes the value of the timestamp captured from the Sync packet. In addition the master should timestamp Delay_Req packets on its RX path and return to the slave that sent it the timestamp value using a Delay_Response packet. A node in Slave state should timestamp every incoming Sync packet and if it came from its selected master keep this value for time offset calculation. In addition it should periodically send Delay_Req packets in order to

calculate the path delay from its master. Every sent Delay_Req packet sent by the slave is time stamped and kept. With the value received from the master with Delay_Response packet the slave can now calculate the path delay from the master to the slave.

The implementation of this protocol is typically distributed between hardware and software.

The HW responsibilities are:

- Identify the packets that require time stamping.
- Time stamp the packets on both RX and TX paths.
- Store the time stamp value for SW.
- Keep the system time in HW and give a time adjustment service to the SW.
- Maintain auxiliary features related to the system time.

The SW responsibilities are:

- BMC protocol execution which means defining the node state (master or slave) and selection of the master clock if in slave state.
- Read time stamps from HW and generate PTP packets.
- Calculate the time offset and adjust the system time using HW mechanism for that.
- Enable configuration and usage of the auxiliary features.

41.5.1.2.1 Protocol for Ordinary and Boundary Clocks

The synchronization protocol flow and the offset calculation for an ordinary or boundary clock model are described in Figure 41-3.

Figure 41-3. Clock Synchronization

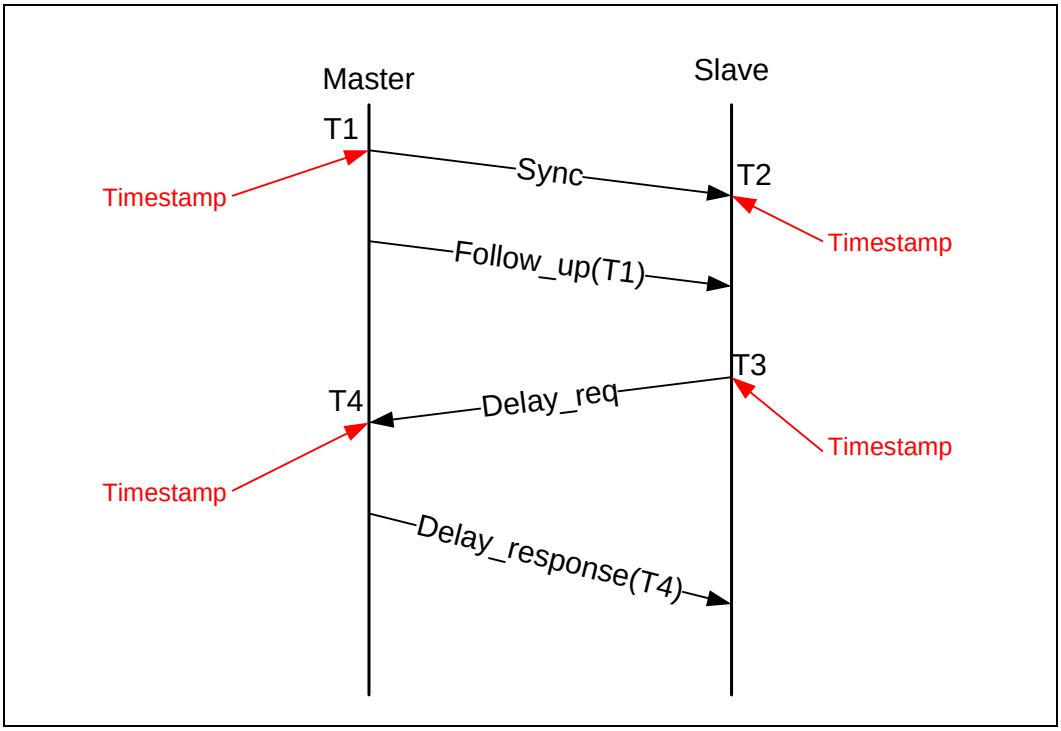


Table 41-2. Clock Synchronization Protocol Flow

Action	Responsibility	Node Type
Generate a Sync Packet	SW	Master
Timestamp the Sync packet and store the value in registers (T1)	HW	Master
Timestamp the incoming Sync packet; store the value in a register (T2); and record the sourceID and sequenceId in registers	HW	Slave
Read the timestamp register (T1) and put the value in a Follow_Up packet and send it.	SW	Master
Note the timestamp (T1) from the received Follow-up message	SW	Slave
Generate a Delay_Req packet and send it	SW	Slave
Timestamp the outgoing Delay_Req packet and store in register (T3)	HW	Slave
Timestamp incoming Delay_Req message; store value (T4); record sourceID and sequenceID in registers	HW	Master
Read timestamp (T4) from register and send back to slave using a Delay_Response packet	SW	Master
Note the timestamp (T4) from received Delay_Resp packet and calculate the time offset using T1, T2, T3, and T4	SW	Slave

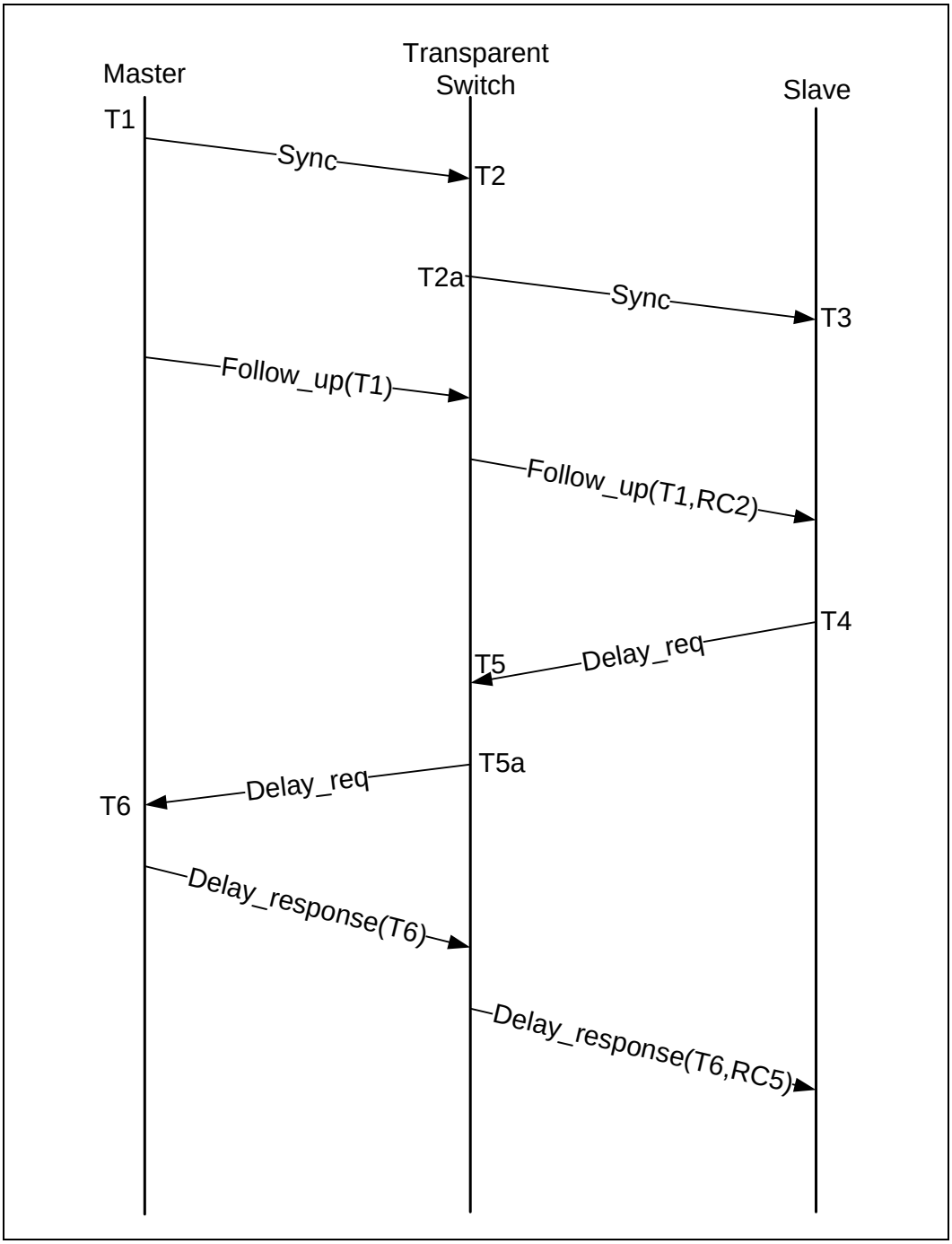
41.5.1.2.2 Protocol for Transparent Switches

The synchronization protocol flow through a transparent switch is described in Figure 41-4.

Table 41-3. Transparent Clock Synchronization Protocol Flow

Action	Responsibility	Node Type
Sync Packet		
Generate a Sync Packet	SW	Master
Timestamp the Sync packet and store the value in registers (T1)	HW	Master
Timestamp the incoming Sync packet; store the value in a register (T2); and record the sourceID and sequenceId in registers	HW	Transparent Switch
Forward the received Sync Packet to the slave	SW	Transparent Switch
Timestamp the Sync packet and store the value in registers (T2a)	HW	Transparent Switch
Timestamp the incoming Sync packet; store the value in a register (T3); and record the sourceID and sequenceId in registers	HW	Slave
Follow_Up Packet		
Read the timestamp register (T1) and put the value in a Follow_Up packet and send it.	SW	Master
Forward the received Follow_Up packet, first appending the Residence Time (RC2) of the previous Sync packet. RC2 being the delay between receiving the Sync packet (T2) and transmitting the sync packet (T2a)	SW	Transparent Switch
Note the timestamp (T1) and the residence time (RC2) from the received Follow-up message	SW	Slave
Delay_Req Packet		
Generate a Delay_Req packet and send it	SW	Slave
Timestamp the outgoing Delay_Req packet and store in register (T4)	HW	Slave
Timestamp incoming Delay_Req message; store value (T5); record sourceID and sequenceID in registers	HW	Transparent Switch
Forward the received Delay_Req Packet to the master	SW	Transparent Switch
Timestamp the outgoing Delay_Req packet and store the value in registers (T5a)	HW	Transparent Switch
Timestamp incoming Delay_Req message; store value (T6); record sourceID and sequenceID in registers	HW	Master
Delay_Response Packet		
Read timestamp (T6) from register and send back to slave using a Delay_Response packet	SW	Master
Forward the received Delay_Response packet, first appending the Residence Time (RC5) of the previous Delay_Req packet. RC5 being the delay between receiving the Delay_Req packet (T5) and transmitting the Delay_Req packet (T5a)	SW	Transparent Switch
Note the timestamp (T6) and the residence time RC5 from received Delay_Resp packet and calculate the time offset using T1, RC2, T3, T4, RC5 and T6	SW	Slave

Figure 41-4. Transparent Clock Switch Protocol Flow



41.5.1.3 PTP Message Formats

The time sync implementation supports both IEEE1588 V1 and IEEE1588-2008 PTP messages. formats The format for each of these is shown in Table 41-4.

When a PTP message containing packet is recognized, the PTP version is checked. If it is V1 PTP message then the control field contains the message type. The message type decoding is shown in Table 41-5 and Table 41-6.

Table 41-4. IEEE1588 Version 1 and IEEE1588-2008 PTP Message Formats

Offset in bytes	V1 Fields	IEEE1588-2008 Fields
0	VersionPTP	Transport Specific/messageId
1		Reserved (4 bits)/VersionPTP(4 bits)
2	VersionNetwork	MessageLength
3		
4	Subdomain	SubdomainNumber
5		Reserved
6		Flags
7		
8		CorrectionNs
9		
10		
11		
12		
13		
14		CorrectionSubNs
15		
16		Reserved
17		
18		
19		
20	MessageType	Reserved
21	Source communication technology	Source communication technology
22	Sourceuuid	Sourceuuid
23		
24		
25		
26		
27		
28	Sourceportid	Sourceportid
29		
30	SequenceID	SequenceID
31		

Table 41-4. IEEE1588 Version 1 and IEEE1588-2008 PTP Message Formats

Offset in bytes	V1 Fields	IEEE1588-2008 Fields
32	Control	Control
33	Reserved	
34	Flags	LogMessagePeriod
35		

Table 41-5. Message Decoding for V1

Enumeration	Value	Note
PTP_SYNC_MESSAGE	0	Time stamp
PTP_DELAY_REQ_MESSAGE	1	Time stamp
PTP_FOLLOWUP_MESSAGE	2	
PTP_DELAY_RESP_MESSAGE	3	
PTP_MANAGEMENT_MESSAGE	4	
reserved	5-255	

Table 41-6. Message decoding for IEEE1588-2008

MessageId	Message Type	Value (hex)	Note
PTP_SYNC_MESSAGE	Event	0	Time stamp
PTP_DELAY_REQ_MESSAGE	Event	1	Time stamp
PTP_PATH_DELAY_REQ_MESSAGE	Event	2	Time stamp
PTP_PATH_DELAY_RESP_MESSAGE	Event	3	Time stamp
Unused		4-7	
PTP_FOLLOWUP_MESSAGE	General	8	
PTP_DELAY_RESP_MESSAGE	General	9	
PTP_PATH_DELAY_FOLLOWUP_MESSAGE	General	A	
PTP_ANNOUNCE_MESSAGE	General	B	
PTP_SIGNALLING_MESSAGE	General	C	
PTP_MANAGEMENT_MESSAGE	General	D	
Unused		E-F	

41.5.2 Time Stamping Operation

Time stamping means that the current value in the system time register is captured in a second register, generally called a snapshot register.

The time stamping will occur when the appropriate conditions exist such as an auxiliary input is received or when a particular type of message is transmitted or received by the channel. Once a timestamp of system time is taken and locked, a unique indication for the snapshot is set in the appropriate Event register. No further timestamps of that

type can be received until the snapshot indication is cleared by firmware. Thus, the setting of the indication is a lockout of further snapshots of a particular type until firmware takes action (unless the traffic analyzer lock inhibit feature is enabled).

41.5.2.1 Sync Messages

The SW for the master channel sends a Sync message periodically over the network at 1-, 2-, 8-, 16-, or 64-second intervals.

If the channel is a master, the Time Sync logic will monitor the interface and detect when a Sync message has been transmitted. When a Sync message is detected and the XMIT_Snapshot is not locked out, the message is time stamped and the current system time is captured in the XMIT_Snapshot register. If the message is transmitted with no errors, the XMIT_Snapshot is locked.

If the channel is a slave, the Time Sync logic will monitor the interface and detect when a Sync message has been received. When the Sync message is detected and the RECV_Snapshot is not locked out, the message is time stamped and the current system time is captured in the RECV_Snapshot register. If the message is received with no errors, the RECV_Snapshot is locked.

When the snapshot of the Sync message has occurred, an indication asserts in the TS_Channel_Event register and remains set until firmware explicitly writes a ‘1’ back to that bit. Until the Sync message snapshot indication is cleared, no further Sync messages will be time stamped. Locking can be inhibited by setting the TS_Channel_Control register appropriately.

41.5.2.2 Follow-up Messages

The Time Sync logic performs no action related to Follow-up messages. It is the responsibility of the SW for the Master to read the XMIT_Snapshot register and send the Follow-up message containing this timestamp.

41.5.2.3 Delay_Req Message

Slave channels transmit a Delay_Req message to the master in response to receiving a Sync message.

If the channel is a master, the Time Sync logic will monitor the interface and detect when a Delay_Req message has been received. When the message is detected and the RECV_Snapshot is not locked out, the message is time stamped and the current system time is captured in the RECV_Snapshot register. If the message is received with no errors, the RECV_Snapshot is locked.

If the channel is a slave, the Time Sync logic will monitor the interface and detect when a Delay_Req message has been transmitted. When the message is detected and the XMIT_Snapshot is not locked out, the message is time stamped and the current system time is captured in the XMIT_Snapshot register. If the message is transmitted with no errors, the XMIT_Snapshot is locked.

When the snapshot for the Delay_Req message has occurred, an indication asserts in the TS_Channel_Event register and remains set until firmware explicitly writes a ‘1’ back to that bit. Until the Delay_Req message snapshot indication is cleared, no further Delay_Req messages will be time stamped. This is important to note since multiple slave channels may try to send Delay_Req messages simultaneously. Locking can be inhibited by setting the TS_Channel_Control register appropriately.

41.5.2.4 Delay_Response Messages

The Time Sync logic performs no action related to Delay_Response messages. It is the responsibility of the SW for the Master to read the RECV_Snapshot register and send the Delay_Response message containing this timestamp.

41.5.2.5 Error Handling

The time synchronization hardware depends on software for filtering appropriate IEEE-1588 packets. When the software receives a Sync or Delay Request message, it interrogates the 1588 hardware for time snapshot information. However before using this data, the software must do several reasonableness checks on the data. This is due to the possibility of lost messages, the limited 1588 queue size of 1 entry, the possibility of multiple domains (the 1588 hardware only supports 1 domain), the asynchronous nature of the software clearing the timestamp lock with respect to possible incoming messages, and other similar events. Therefore the hardware includes some mechanisms to assist the software filtering, such as the capture of the UUID and Sequence Count.

When a time sync message is received by software that is expected to have an associated timestamp, the software should perform the following checks:

1. Compare snapshot UUID to UUID in received 1588 packet.
2. Compare snapshot Sequence Count to Sequence Count in received 1588 packet.
3. Verify that timestamp is different than last timestamp.

If any of these tests fail, the software should clear the lock and discard the information.

41.5.3 IEEE1588 over Ethernet

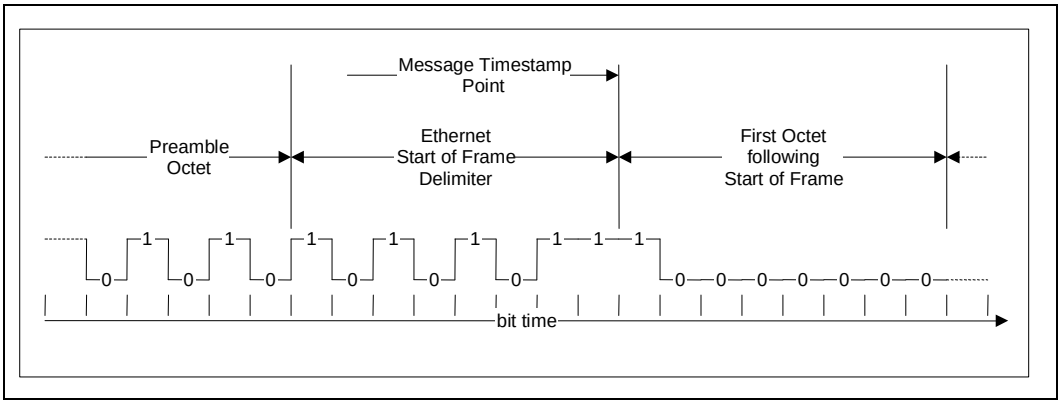
41.5.3.1 Timestamping Mechanism

Per the 1588 specification, synchronized time is referenced to the end of the “start of frame delimiter” (SFD) as shown Figure 41-5.

The time sync hardware captures the system time immediately upon detection of the SFD. The timestamp point is immediately after the SFD. This timestamp is stored in the snapshot register and is frozen in the snapshot register when the last nibble of the frame CRC is transmitted or received and the overall message is detected with no errors. If an errored frame is detected (TX_ERR or RX_ERR are asserted) then the frame/message will be ignored.

Due to PHY and synchronization delays, the actual timestamp will be slightly later than the desired reference point. However, allowing for 1 `pclk` synchronization jitter, this is a fixed delay, easily nulled out in the software portion of the algorithm. This fixed delay is dependent on the 10/100/1000 MHz selection at the PHY. Therefore, a constant can be subtracted from the snapshot to compensate for PHY and synchronization delays to arrive at the IEEE-1588 specified time stamp point.

Figure 41-5. Time Stamp Reference Point



41.5.3.2 PTP Message Detection in Ethernet Frames

IEEE1588 PTP messages may be detected within three types of ethernet frames.

- L4 UDP frames where the port is of the type “EventPort”
- L2 Ethernet frames where the EtherType is defined to be “IEEE1588”
- A user defined frame in which configurable offsets, masks and compare values can be defined by the user via registers. This mode is provided to facilitate usage with a custom or otherwise non-IEEE1588 compliant system.

The specific decoding to identify a PTP containing frame is shown in Table 41-7.

Table 41-7. PTP Frame Identification

Field	Field Size	L4	L2	User Defined
Ethernet Header				
Dest MAC Address	6	x	x	x
Source MAC Address	6	x	x	x
EtherType	2	0x0800 --- OR --- 0x8100 (indicates tagged frame)	Programmable value (default = 0x88f7 IEEE1588) --- OR --- 0x8100 for tagged frame	Programmable mask and value
VLAN Header				
VLAN tag control (only present if EtherType = 0x8100)	2	Ignore if present	Ignore if present	Not Present
EtherType (only present in tagged frame)	2		If tagged frame then test against programmed value (default = 0x88f7)	
IP Header				

Table 41-7. PTP Frame Identification

Field	Field Size	L4	L2	User Defined
Version	1	0x45 (IPv4)	Not Present	Not Present
TOS	1	x		
Length	2	x		
ID	2	x		
Flags/Frag Offset	2	x		
TTL	1	x		
Protocol	1	0x11 (UDP)		
Header Checksum	2	x		
Source Address	4	x		
Dest Address	4	x		
UPD Header				
Source port	2	0x013F (EventPort type)	Not Present	Not Present
Dest port	2	x		
Length	2	x		
Checksum	2	x		
PTP Header **See Table 41-4, "IEEE1588 Version 1 and IEEE1588-2008 PTP Message Formats" on page 1628**				**User defined** Match programmable location against a programmable value to determine to timestamp

41.5.3.3 Modes of Operation

The specific message detection mode for each channel can be configured. Table 41-8 documents the supported modes of operation.

Table 41-8. Timestamping Configurations

"Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)" on page 1656				Behavior					
Version [31]	Mode [20:16]	mm [0]	ta [1]	L2	L4	PTP Version 1	PTP Version 2	Messages	Locked
0	ignore	0	0	No	Yes	Yes	No	Sync (Rx Only) Delay_req (Tx Only)	Yes
0	ignore	1	0	No	Yes	Yes	No	Sync (Tx Only) Delay_req (Rx Only)	Yes
0	ignore	ignore	1	No	Yes	Yes	No	Sync Delay_req	No
1	0	ignore	ignore	No	Yes	Yes	No	Sync Delay_req	Yes
1	1	ignore	ignore	No	Yes	Yes	No	All messages	No

Table 41-8. Timestamping Configurations

"Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)" on page 1656				Behavior					
Version [31]	Mode [20:16]	mm [0]	ta [1]	L2	L4	PTP Version 1	PTP Version 2	Messages	Locked
1	2	ignore	ignore	Yes	Yes	Yes	Yes	**All Event** Sync Delay_REQ Path_Delay_Req Path Delay_Resp	Yes
1	3	ignore	ignore	Yes	Yes	Yes	Yes	All messages	No
1	7:4	Reserved							
1	8	ignore	ignore	ignore	ignore	ignore	ignore	User-defined mode	Yes

When the mode of operation is "Locked", the timestamp taken after the SFD is frozen in the snapshot registers and will not be updated until the software reset it.

When the mode of operation is not "Locked", Each message is time stamped at the reception of a start of frame delimiter (SFD), however the snapshot registers will be overwritten with the arrival of a subsequent PTP message.

41.5.4 IEEE1588 over CAN

The time synchronization logic supports a hardware assist implementation for a CAN network (e.g. DeviceNet). The 1588 protocol operates over a CAN network in much the same way as it does over Ethernet, using the same time synchronization messages identified earlier. However, the CAN protocol requires that these relatively lengthy messages are broken into much smaller frames. This fragmentation prevents one device from utilizing excessive bandwidth and maintains real time access to the network for all devices.

Since the CAN protocol breaks up 1588 messages into small frames and the frames do not carry 1588 specific identifiers, it is not practical to identify the 1588 Sync and Delay Request messages in hardware. Therefore, the hardware merely captures a timestamp into a holding register at the appropriate point in each and every frame that is transmitted or received. The software has the responsibility to log the captured timestamp as part of each frame, as the software processes the transmit done interrupt (at the completion of a sent frame) or the received frame ready interrupt (at the availability of a received frame).

This hardware assisted approach eliminates the potential for variable software interrupt service times from introducing jitter in the time snapshot and provides improved accuracy over a software-only approach. However, it does require significant software support. For example, when using 1588 hardware support for CAN, the multiple message buffers and screeners of the normal CAN block cannot be used in a continually over-writing updating mode without processor intervention. This is due to the fact that the software must read and save the time snapshot before the next message is sent or received. Therefore, when 1588 hardware support is used, software will only be able to effectively use 1 screener/buffer and must guarantee that all CAN frame interrupts are

serviced, which may come as often as the minimum frame size. These requirements should be easily met since CAN is a half duplex protocol operating at a relatively low baud rate.

When a CAN packet is received or transmitted, the interrupt signal from the CAN device will assert and cause a snapshot of system time to be captured in the TS_CanSnapLo and TS_CanSnapHi registers. The TS_CanSnapLo register contains the lower 32 bits of the time value, and the TS_CanSnapHi register contains the upper 32 bits. The interrupt signal from the CAN must be enabled to assert on receive and transmit completion of each packet in order to capture snapshots for CAN packets. It is up to the firmware to assemble the packets into messages and determine which packet's snapshot is the appropriate one for the entire message. There is one pair of CAN snapshot registers (low and high) for each CAN device.

Therefore, for each CAN device, the assertion of the CAN interrupt signal when a frame transmit or receive is completed will capture the system time in a 64-bit snapshot register for that CAN device. Each frame that is received or transmitted will have a snapshot taken. Firmware will process the frames as part of the overall message evaluation, identify valid time sync messages, and determine the appropriate snapshot to utilize. Two CAN channels are currently supported.

Note: In order to Timestamp CAN activity the following assumptions must be valid.

- The CAN interrupts from the CAN controller MUST be serviced by Software before the next CAN frame is detected. The initial assertion of the interrupt signal will initiate a time snapshot of the first frame but no further snapshots will be taken and no 1588 overflow will be set on any subsequent frames. However, this condition is detected, not by the 1588 block but by the CAN controller itself. The primary detection of a missed CAN interrupt is still (and always was) an overrun condition in the CAN controller.
- In1588 mode, it is expected that only one of the 16 available CAN receive buffers will be enabled and used. The relatively slow speed of the CAN channel makes this an acceptable trade-off between hardware and performance. When a frame is received, the CAN controller sets the buffer's MsgAv bit. If another frame is received and no receive buffers are available (as would be the case if only one buffer was enabled and its MsgAv bit was set from a previous frame not yet serviced), the RxMsgLost flag is set in the CAN IP. Thus the software would recognize this overflow condition by the RxMsgLost flag when it finally got around to servicing the interrupt.

41.5.5 Auxiliary Snapshots

Time stamps may also be taken based on externally provided signals (asmssig and/or ammssig). These time stamps are recorded in the auxiliary snapshot registers.

When the signal is asserted the hardware will take a timestamp and set the lock. Software may then be notified via interrupt (if so configured) and it will read the appropriate register. If the software clears the lock before the signal is de-asserted, then a second, redundant snapshot event will be generated.

Note: Hardware filtering and edge-detection were considered but not implemented because the signal quality from the master could be bad enough to cause spurious locks. For example, cables to a GPS could be a kilometer or more in length and the type of cable could be a factor as well. The firmware handles the filtering and MUST not clear the lock until after the master has negated the snapshot input.

41.5.6 Target Time Expiration

A system time that is greater than or equal to the target time sets the target time expiration condition and generates an interrupt if enabled.

Two independent target times are supported.

Refer to Section 41.6.1.11, “Offset 0028h: TS_TrgtLo - Target Time Low Register” and Section 41.6.1.12, “Offset 002Ch: TS_TrgtHi - Target Time High Register”.

Also, Section 41.6.1.28, “Offset 01F0h: TS_Aux_TrgtLo - Auxiliary Target Time Low Register” and Section 41.6.1.29, “Offset 01F4h: TS_Aux_TrgtHi -Auxiliary Target Time High Register”

41.5.7 System Time

The system timer is derived from the source clock through a frequency divider. The frequency divider is based on the value in the Addend register. Once every system clock cycle the value in the Addend register is added to the value in the Accumulator register. When the accumulator register rolls over, the system timer is incremented.

A single SourceClock is provided from which the desired SystemClock can be derived via a frequency divider. The frequency divider is implemented by using the Addend register. Once every SourceClock cycle the value in the Addend register is added to the value in the Accumulator register. When the accumulator register rolls over, the system timer is incremented. Thus, the system timer is incremented once every SystemClock period.

The value in the Addend register represents the frequency compensation value. This value is determined as follows:

FreqDivisionRatio = FreqOscillator / FreqClock

The equation for the frequency compensation value utilizes the precision of the accumulator and the FreqDivisionRatio. Since the accumulator is 32 bits, the following equation applies:

$$FreqCompensationValue = 2^{32} / FreqDivisionRatio$$

The hexadecimal representation of the FreqCompensationValue is the value that is written to the Addend register. The following table gives examples of addend values based on a 66.67 MHz FreqOscillator:

Table 41-9. Addend Values

FreqOscillator[1]	FreqClock	FreqDivisionRatio	FreqCompensationValue
66.67 MHz	40 MHz	1.67	0x994B1D20
66.67 MHz	50 MHz	1.33	0xC07B301E
66.67 MHz	60 MHz	1.11	0xE6A17102

1. 50 MHz for 600 MHz IA-32 core and 66.67 MHz for 1066 MHz/1200 MHz IA-32 core.

41.5.8 Interrupts

An level sensitive interrupt signal to the processor is generated when any of the following conditions occur and are enabled:

- Auxiliary Master Mode snapshot is taken
- Auxiliary Slave Mode snapshot is taken
- Target time expiration.
- Auxiliary Target Time expiration
- Pulse per Second assertion

An interrupt enable mask must be set to allow any of the interrupts to pass to the core.

41.5.9 Reset

There are two types of reset:

- Power-on reset
- MMR write to soft reset register to initiate a channel

Each channel may be independently reset by SW by writing to the channel's control register.

If the GMII interface changes clock speeds (1000Mbps down to 100/10 Mbps), SW must initiate a soft reset to the corresponding channel.

41.6 Register Summary

Writes to unused address space will have no affect. Reads to unused address space may return indeterminate data. Neither situation should cause detrimental effects on device operation unless specifically documented. Reserved bits within registers must be written with their reset value unless otherwise stated.

For more information on the conventions the following register summaries adopt, see Section 7.1, "Overview of Register Descriptions and Summaries" on page 183.

The IEEE 1588 Hardware Assist registers materialize in the PCI space

Table 41-10 summarizes the IEEE 1588 TSYNC materialization from the PCI perspective.

Table 41-10. Bus M, Device 7, Function 0: Summary of IEEE 1588 TSYNC CSRs (Sheet 1 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"Offset 0000h: TS_Control Register" on page 1639	00000000h
00000004h	00000007h	"Offset 0004h: TS_Event Register" on page 1641	0022h
00000008h	0000000Bh	"Offset 0008h: TS_Addend Register" on page 1643	0000h
0000000Ch	0000000Fh	"Offset 000Ch: TS_Accum Register" on page 1643	0000h
00000010h	00000013h	"Offset 0010h: TS_Test Register" on page 1644	0000h
00000014h	00000017h	"Offset 0014h: TS_PPS_Compare Register" on page 1646	FFFFFFFFh

Table 41-10. Bus M, Device 7, Function 0: Summary of IEEE 1588 TSYNC CSRs (Sheet 2 of 2)

Offset Start	Offset End	Register ID - Description	Default Value
00000018h	0000001Bh	"Offset 0018h: TS_RSysTimeLo Register" on page 1647	0000h
0000001Ch	0000001Fh	"Offset 001Ch: TS_RSysTimeHI Register" on page 1648	0000h
00000020h	00000023h	"Offset 0020h: TS_SysTimeLo Register" on page 1649	0000h
00000024h	00000027h	"Offset 0024h: TS_SysTimeHi Register" on page 1650	0000h
00000028h	0000002Bh	"Offset 0028h: TS_TrgtLo Register" on page 1650	0000h
0000002Ch	0000002Fh	"Offset 002Ch: TS_TrgtHi Register" on page 1651	0000h
00000030h	00000033h	"Offset 0030h: TS_ASMSLo Register" on page 1652	0000h
00000034h	00000037h	"Offset 0034h: TS_ASMSHi Register" on page 1653	0000h
00000038h	0000003Bh	"Offset 0038h: TS_AMMSLo Register" on page 1654	0000h
0000003Ch	0000003Fh	"Offset 003Ch: TS_AMMSHi Register" on page 1655	0000h
0040h at 20h	0043h at 20h	"Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)" on page 1656	0000h
0044h at 20h	0047h at 20h	"Offset 0044h: TS_CH_EVENT[0-7] - Time Synchronization Channel Event Register Per Ethernet Channel)" on page 1658	0000h
0048h at 20h	004Bh at 20h	"Offset 0048h: TS_TxSnapLo[0-7] - Transmit Snapshot Low Register (Per Ethernet Channel)" on page 1659	0000h
004Ch at 20h	004Fh at 20h	"Offset 004Ch: TS_TxSnapHi[0-7] - Transmit Snapshot High Register (Per Ethernet Channel)" on page 1660	0000h
0050h at 20h	0053h at 20h	"Offset 0050h: TS_RxSnapLo[0-7] - Receive Snapshot Low Register (Per Ethernet Channel)" on page 1661	0000h
0054h at 20h	0057h at 20h	"Offset 0054h: TS_RxSnapHi[0-7] - Receive Snapshot High Register (Per Ethernet Channel)" on page 1662	0000h
0058h at 20h	005Bh at 20h	"Offset 0058h: TS_SrcUUIDLo[0-7] - Source UUID0 Low Register (Per Ethernet Channel)" on page 1663	0000h
005Ch at 20h	005Fh at 20h	"Offset 005Ch: TS_SrcUUIDHI[0-7] - SequenceID/SourceUUID High Register (Per Ethernet Channel)" on page 1664	0000h
0140h at 10h	0143h at 10h	"Offset 0140h: TS_CANx_Status[0-1] - Time Synchronization Channel Event Register (Per CAN Channel)" on page 1665	0000h
0144h at 10h	0147h at 10h	"Offset 0144h: TS_CANSnapLo[0-1] - Transmit Snapshot Low Register (Per CAN Channel)" on page 1666	0000h
0148h at 10h	014Bh at 10h	"Offset 0148h: TS_CANSnapHi[0-1] - Transmit Snapshot High Register (Per CAN Channel)" on page 1667	0000h
000001F0h	000001F3h	"Offset 01F0h: TS_Aux_TrgtLo Register" on page 1668	0000h
000001F4h	000001F7h	"Offset 01F4h: TS_Aux_TrgtHi Register" on page 1668	0000h
00000200h	00000203h	"Offset 0200h: L2 EtherType Register" on page 1669	000088F7h
0000204h	0000207h	"Offset 0204h: User Defined EtherType Register" on page 1669	00000000h
00000208h	0000020Bh	"Offset 0208h:User Defined Header Offset Register" on page 1670	00000000h
0000020Ch	0000020Fh	"Offset 020Ch:User Defined Header Register" on page 1670	00000000h

41.6.1 Detailed Register Descriptions

41.6.1.1 Offset 0000h: TS_Control - Time Sync Control Register

Register Name	TS_Control																Access		(See below.)				Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)																										atm	ppsm	amm	asm	ttm	rst

Table 41-11. Offset 0000h: TS_Control Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000000h **Offset End:** 00000003h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 6	RSVD	Reserved for future use. Must be written as '0'		0	RO
5 : 5	atm	**Auxiliary Target Time Interrupt Mask.** The AuxiliaryTarget Time interrupt mask controls whether the Auxiliary Target Time interrupt is passed to the Host processor. When this bit is set, the interrupt to the Host is enabled. When cleared, the Auxiliary Target Time interrupt to the Host is disabled.		0h	RW
4 : 4	ppsm	PPS Interrupt Mask. The PPS interrupt mask controls whether the 1 PPS Compare register match indication, which is the **pps** bit in the Time Sync Event register, should interrupt the Host processor. When this bit is set, the interrupt to the Host is enabled. When cleared, the PPS interrupt to the Host is disabled.		0h	RW
3 : 3	amm	**AMMS Interrupt Mask.** Controls whether the Auxiliary Master Mode snapshot indication, which is the snm bit in the Time Sync Event register, should interrupt the Host processor. • When this bit is set, the interrupt to the Host is enabled. • When cleared, the AMMS interrupt to the Host is disabled.		0h	RW

Table 41-11. Offset 0000h: TS_Control Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000000h **Offset End:** 00000003h	
Size: 32 bits	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
2 : 2	asm	**ASMS Interrupt Mask.** Controls whether the indication that an Auxiliary Slave Mode snapshot, which is the sns bit in the Time Sync Event register, has been taken should interrupt the Host processor. • When this bit is set, the interrupt to the Host is enabled. • When cleared, the ASMS interrupt to the Host is disabled.		0h	RW
1 : 1	ttm	**Target Time Interrupt Mask.** Controls whether the Target Time interrupt is passed to the Host processor. • When this bit is set, the interrupt to the Host is enabled. • When cleared, the Target Time interrupt to the Host is disabled.		0h	RW
0 : 0	rst	Reset. • When a ‘1’ is written to this bit, all logic is returned to the same default state as when a power-on reset occurs. • After writing a ‘1’ to this bit to reset the logic, the firmware must write a ‘0’ to the bit to indicate the end of the reset.		0h	RW

41.6.1.2 Offset 0004h: TS_Event - Time Sync Event Register

Register Name	TS_Event																Access		(See below.)		Reset Value				0x0000_0022			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)																						gbe1_mode	gbe0_mode	(reserved)		atp	pps	snm	sns	ttipend	rsvd

Table 41-12. Offset 0004h: TS_Event Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000004h **Offset End:** 00000007h	
Size: 32 bits	**Default:** 0022h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 10	RSVD	Reserved for future use.		x	RO
9 : 9	gbe1_mode	GBe1_mii_mode status: “1” indicates the gbe 1 port is running in MII mode “0” indicates the gbe 1 port is running in GMII mode		0h	RO
8 : 8	gbe0_mode	GBe0_mii_mode status: “1” indicates the gbe 1 port is running in MII mode “0” indicates the gbe 1 port is running in GMII mode		0h	RO
7 : 6	RSVD	Reserved for future use.		0h	RO
5 : 5	atp	**Auxiliary Target Time Interrupt Pending.** This bit is the Auxiliary Target Time interrupt pending indication. When this bit is set, it indicates that the Auxiliary Target Time interrupt condition has occurred, which means that the System Time value has reached the 64-bit Auxiliary Target Time register value. If **atm** in the Time Sync Control register is set, the interrupt will be passed to the Host processor. To clear this condition and also the interrupt to the Host if no other sources are driving it, the firmware must write a ‘1’ to the **atp** bit. To prevent an immediate reoccurrence of the auxiliary target time interrupt, the processor should first write a new value to the Auxiliary Target Time register and then clear the condition. This bit is set at power-up since both the System Time and the Auxiliary Target Time are reset at power-up to 0x0000000000000000.		1	RWC
4 : 4	pps	**PPS Match.** This event bit sets when the lower 32 bits of the system time register is equal to the 1PPS Compare register. When this signal is asserted high, an interrupt will be generated to the Host on the ts_intreq if the ppsm bit in the Time Sync Control register is also set. This signal also drives the ts_pps output pin of the TimeSync block. The user will clear pps by writing a '1' to it.		0h	RWC
3 : 3	snm	**AMMS Snapshot.** This event bit sets when the system time register value is captured in the Auxiliary Master Mode Snapshot register upon an active high level on a general purpose input, ammssig. • When this signal is asserted high, an interrupt will be generated to the Host on the ts_intreq if the amm bit in the Time Sync Control register is also set. • To clear snm, write a ‘1’ to it.		0h	RWC

Table 41-12. Offset 0004h: TS_Event Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000004h **Offset End:** 00000007h	
Size: 32 bits	**Default:** 0022h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
2 : 2	sns	**ASMS Snapshot.** This event bit sets when the system time register value is captured in the Auxiliary Slave Mode Snapshot register upon detection of a active high level on a general purpose input, asmssig. • When this signal is asserted high, an interrupt will be generated to the Host on the shared interrupt signal (ts_ntreq) if the asm bit in the Time Sync Control register is set. • To clear the sns bit, write a '1' to it.		0h	RWC
1 : 1	ttipend	**Target Time Interrupt Pending.** This bit is the Target Time interrupt pending indication. When this bit is set, it indicates that the Target Time interrupt condition has occurred, which means that the System Time value has reached the 64-bit Target Time register value. • If ttm in the Time Sync Control register is set, the interrupt will be passed to the Host processor. • To clear this condition, the firmware must write a '1' to the ttipend bit. To prevent an immediate reoccurrence of the target time interrupt, the processor should first write a new value to the Target Time register and then clear the condition. This bit is set at power-up since both the System Time and the Target Time are reset at power-up to 0.		1	RWC
0 : 0	RSVD	Reserved for future use.		0h	RO

41.6.1.3 Offset 0008h: TS_Addend - Addend Register

Register Name	TS_Addend																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Addend[31:0]																															

Table 41-13. Offset 0008h: TS_Addend Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000008h **Offset End:** 0000000Bh	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	Addend	The Addend register contains the frequency scaling value used by a firmware algorithm to achieve time synchronization in the module. The value in this register is added to the value in the Accumulator. When the Accumulator rolls over, an overflow pulse is asserted and increments system time. Because the Addend register is cleared at reset, it must be written with a non-zero value to allow system time to increment.		0h	RW

41.6.1.4 Offset 000Ch: TS_Accum - Accumulator Register

Register Name	TS_Accum																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Accumulator[31:0]																															

Table 41-14. Offset 000Ch: TS_Accum Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0000000Ch **Offset End:** 0000000Fh	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	Accumulator	The Accumulator register serves as the frequency divider in the time synchronization logic. Firmware calculates a frequency scaling value to be written to the Addend register. The data in the Accumulator register is added to the value in the Addend register once every period of the system clock. When the Accumulator rolls over, an overflow pulse is asserted which increments the value in the system timer. This register is not read or written to in normal operation.		0000h	RW

41.6.1.5 Offset 0010h: TS_Test - Time Sync Test Register

Register Name	TS_Test																Access	(See below.)			Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)																				rx_snap_cfg				tx_snap_cfg				tenb			tm

Table 41-15. Offset 0010h: TS_Test Register (Sheet 1 of 2)

Description:					
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:7:0		**Offset Start:** 00000010h **Offset End:** 00000013h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 12	RSVD	Reserved for future use.		x	RV
11 : 8	rx_snap_cfg	Determines which RX channel activity is reflected on TS_RX_SNAP_TAKEN 0000 -- Channel 0 RX snapshot activity monitored 0001 -- Channel 1 RX snapshot activity monitored 0010 -- Channel 2 RX snapshot activity monitored 0011 -- Channel 3 RX snapshot activity monitored 0100 -- Channel 4 RX snapshot activity monitored 0101 -- RESERVED for Channel 5 RX snapshot activity monitored 0110 -- RESERVED for Channel 6 RX snapshot activity monitored 0111 -- RESERVED for Channel 7 RX snapshot activity monitored 1000 -- CAN 0 snapshot activity monitored 1001 -- CAN 1 snapshot activity monitored 1010 -1111 -- RESERVED ***Note:*** The ts_rx_snap_taken signal is non-zero when the tm bit is set and will be a pulse that is eight pclk cycles wide.		0000h	RW
7 : 4	tx_snap_cfg	Determines which TX channel activity is reflected on TS_TX_SNAP_TAKEN 0000 -- Channel 0 TX snapshot activity monitored 0001 -- Channel 1 TX snapshot activity monitored 0010 -- Channel 2 TX snapshot activity monitored 0011 -- Channel 3 TX snapshot activity monitored 0100 -- Channel 4 TX snapshot activity monitored 0101 -- RESERVED for Channel 5 TX snapshot activity monitored 0110 -- RESERVED for Channel 6 TX snapshot activity monitored 0111 -- RESERVED for Channel 7 TX snapshot activity monitored 1000 -- CAN 0 snapshot activity monitored 1001 -- CAN 1 snapshot activity monitored 1010 -1111 -- RESERVED ***Note:*** The ts_tx_snap_taken signal is non-zero when the tm bit is set and will be a pulse that is eight pclk cycles wide.		0000h	RW

Table 41-15. Offset 0010h: TS_Test Register (Sheet 2 of 2)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000010h **Offset End:** 00000013h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
3 : 1	tenb	**Test Enable.** These bits define what signals drive the ts_testmode_data pin when the tm bit in this register is set. The target time interrupt pending signal (readable in the TS_Event register) is driven if tenb[2:0] is '000' to support future applications. Specific system timer bits drive ts_testmode_data for the remaining settings of tenb[2:0]. tenb[2:0] ts_testmode_data source 000 TS_Event.ttipend 001 TS_SysTimeLo[10] 010 TS_SysTimeLo[12] 101 TS_SysTimeLo[14] 100 TS_Event.auxttipend 101-111 RESERVED		000h	RW
0 : 0	tm	**Test Mode.** This bit, which defaults to '0' at reset, is the test mode bit. • When this bit is set, the IEEE1588 Hardware Assist logic outputs one of four possible signals on the ts_testmode_data pin. The tenb[2:0] bits select the data. This data appears on the ts_testmode_data pin		0h	RW

41.6.1.6 Offset 0014h: TS_PPS - PPS Compare Register

Register Name	TS_PPS_Compare																Access	(See below.)							Reset Value				0xFFFF_FFFF			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPS_Compare Value[31:0]																															

Table 41-16. Offset 0014h: TS_PPS_Compare Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 00000014h Offset End: 00000017h	
Size: 32 bits	Default: FFFFFFFFh			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	TS_PPS_Compare	**The PPS Compare register** is a 32-bit register that contains a value that will be compared against the lower 32 bits of system time. The value placed in this register defines the value of the lower 32 bits of system time required to generate a 1 pulse per second signal to an external scope. When the two values are equal, the pin ts_pps is asserted Concurrently, the state of the signal is visible as the pps bit in the TS_Event register. The pps signal can also interrupt the Host if the ppsm bit in the TS_Channel register is set. It is the firmware's responsibility to calculate the new compare value for the next pulse per second and update this register accordingly. The bits of this register are set at reset in order to prevent ts_pps from asserting right after reset.		FFFFFFFFh	RW
Note: The PPS output controlled by this compare register is independent of the TM bit in the TS_TEST register					

41.6.1.7 Offset 0018h: TS_TSysTimeLo - Raw System Time Low Register

Register Name	TS_RSysTimeLo																Access		(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RawSystemTime_Low[31:0]																															

Table 41-17. Offset 0018h: TS_RSysTimeLo Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000018h **Offset End:** 0000001Bh	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	RawSystemTime_Low	This system time register is a read-only register of the raw system time. It is, therefore, not loadable and reflects the local time in the module. • The lower 32 bits of the 64-bit system time are read in this register. • The upper 32 bits are read in the RawSystemTime_High register. When a user reads system time with this pair of registers, no latching of system time occurs, which means that the system time could increment between the reading of the lower 32 bits in this register and the upper 32 bits in the RawSystemTime_High register. The user must account for this and deal with possible increments between reads of the two registers in firmware.		0000h	RO

41.6.1.8 Offset 001Ch: TS_RSysTimeHI - Raw System Time High Register

Register Name	TS_RSysTimeHI																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RawSystemTime_High[31:0]																															

Table 41-18. Offset 001Ch: TS_RSysTimeHI Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 0000001Ch Offset End: 0000001Fh	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	RawSystemTime_High	This register contains the upper 32 bits of system time. When you want to read or write the system time, this register typically first accesses the RawSystemTime_Low Register. This register pair contains the raw system timer value, and no latching of system time occurs when the lower half is read. Time could increment between the reading of the lower 32 bits in the RawSystemTime_Low register and the reading of this register.		0000h	RO

41.6.1.9 Offset 0020h: TS_SysTimeLo - System Time Low Register

Register Name	TS_SysTimeLo																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SystemTime_Low[31:0]																															

Table 41-19. Offset 0020h: TS_SysTimeLo Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000020h **Offset End:** 00000023h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	SystemTime_Low	The system timer is a loadable up-counter, and reflects the local time in the module. While the system timer is 64 bits wide, the lower 32 bits reside in this register. The system timer is clocked by the module system clock and incremented when the Accumulator register rolls over. To read the entire system time value, the user must read this location first. Reading this location captures the upper 32 bits of the system time in a temporary register, which is accessed when the user reads the SystemTime_High Register next. Likewise, the SystemTime_Low Register must be written first when the user wants to write a new 64-bit value to system time. The data written to this register is captured in a holding register. When the user writes to the SystemTime_High Register, all 64 bits are then written to the system timer. Updating the system time with a direct write has precedence over increments to the system time based on an Accumulator rollover.		0000h	RW

41.6.1.10 Offset 0024h: TS_SysTimeHi - System Time High Register

Register Name	TS_SysTimeHi																Access	(See below.)							Reset Value	0x0000_0000							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SystemTime_High[31:0]																															

Table 41-20. Offset 0024h: TS_SysTimeHi Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 00000024h Offset End: 00000027h	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 0	SystemTime_High	This register contains the upper 32 bits of system time. When the user wants to read or write the system time, this register must first access the SystemTime_Low Register. See “System Time Low Register” on page 4784 for more details.		0000h	RW

41.6.1.11 Offset 0028h: TS_TrgtLo - Target Time Low Register

Register Name	TS_TrgtLo																Access	(See below.)							Reset Value	0x0000_0000							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TargetTime_Low[31:0]																															

Table 41-21. Offset 0028h: TS_TrgtLo Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 00000028h Offset End: 0000002Bh	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 0	TargetTime_Low	The Target Time register set contains 64 bits of a time value. When the system time is greater than or equal to the target time value, an interrupt is generated to the Host on the **ts_intreq** signal if the **ttm** bit in the Time Sync Control register is set. For more information about the Target Time interrupt, see “Time Sync Control Register” on page 4775.		0000h	RW

41.6.1.12 Offset 002Ch: TS_TrgtHi - Target Time High Register

Register Name	TS_TrgtHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TargetTime_Low[31:0]																															

Table 41-22. Offset 002Ch: TS_TrgtHi Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0000002Ch **Offset End:** 0000002Fh	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	TargetTime_High	The Target Time register set contains 64 bits of a time value. When the system time is greater than or equal to the target time value, an interrupt is generated to the Host on the **ts_intreq** signal if the **ttm** bit in the Time Sync Control register is set. For more information about the Target Time interrupt, see Section 102.8.2.1, “Time Sync Control Register” on page 4775.		0000h	RW

41.6.1.13 Offset 0030h: TS_ASMLo - Auxiliary Slave Mode Snapshot Low Register

Register Name	TS_ASMSLo																Access		(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASMS_Low[31:0]																															

Table 41-23. Offset 0030h: TS_ASMSLo Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000030h **Offset End:** 00000033h	
Size: 32 bits	**Default:** **0000**h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	ASMS_Low	When the board is operating in Slave mode, the active high level of a general-purpose input, **asmssig**, triggers a snapshot of System Time into the `ASMS_Low` and `ASMS_High` registers. Note: The processor can configure the GPIO bit as an output, but it will always be input-only to the Time Sync block. When the ASMS snapshot occurs, the **sns** indication in the Time Sync Event register is set. Writing a logic 1 to that bit clears the snapshot indication and allows a new snapshot to occur on the next active high level of **asmssig**.		0000h	RO

41.6.1.14 Offset 0034h: TS_ASMHi - Auxiliary Slave Mode Snapshot High Register

Register Name	TS_ASMSHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASMS_High[31:0]																															

Table 41-24. Offset 0034h: TS_ASMSHi Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 00000034h **Offset End:** 00000037h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	ASMS_High	When the board is operating in Slave mode, the active high level of a general-purpose input, asmssig, triggers a snapshot of System Time into the ASMS_Low and ASMS_High registers. The general-purpose input is synchronized by the Time Sync logic before it is used. Note: The processor can configure the GPIO bit as an output, but it will always be input-only to the Time Sync block. When the ASMS snapshot occurs, the **sns** indication in the Time Sync Event register is set. Writing a logic 1 to that bit clears the snapshot indication and allows a new snapshot to occur on the next active high level of **asmssig**.		0000h	RO

41.6.1.15 Offset 0038h: TS_AMMSLo - Auxiliary Master Mode Snapshot Low Register

Register Name	TS_AMMSLo																Access	(See below.)							Reset Value	0x0000_0000							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMMS_Low[31:0]																															

Table 41-25. Offset 0038h: TS_AMMSLo Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 00000038h Offset End: 0000003Bh	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	AMMS_Low	When the board is operating in Master mode, it receives a general-purpose input signal for synchronization of snapshots and time. This general-purpose input, **ammssig**, is synchronized by the system clock in the Time Sync logic before it is used. **Note:** The processor can configure the GPIO as an output, but it will always be an input-only to the Time Sync block. When the AMMS snapshot occurs, the **snm** indication in the Time Sync Event register is asserted. No new snapshots in the AMMS register pair are captured until the firmware writes a '1' back to the **snm** bit to clear the snapshot indication.		0000h	RO

41.6.1.16 Offset 003Ch: TS_AMMSHi - Auxiliary Master Mode Snapshot High Register

Register Name	TS_AMMSHi																Access		(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMMS_High																															

Table 41-26. Offset 003Ch: TS_AMMSHi Register

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0000003Ch **Offset End:** 0000003Fh	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	AMMS_High	When the board is operating in Master mode, it receives a general-purpose input signal for synchronization of snapshots and time. This general-purpose input, **ammssig**, is synchronized by the system clock in the Time Sync logic before it is used. ***Note:*** The processor can configure the GPIO as an output, but it will always be an input-only to the Time Sync block. When the AMMS snapshot occurs, the **snm** indication in the Time Sync Event register is asserted. No new snapshots in the AMMS register pair are captured until the firmware writes a '1' back to the **snm** bit to clear the snapshot indication.		0000h	RO

41.6.1.17 Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)

Register Name	TS_Ch_Control	Access	(See below.)	Reset Value	x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												crst	ta	mm	

*Address offsets per channel...
Channel 0 = 0x040
Channel 1 = 0x060
Channel 2 = 0x080
Channel 3 = 0x0A0
Channel 4 = 0x0C0
Channel 5 = 0x0E0
Channel 6 = 0x100
Channel 7 = 0x120

Table 41-27. Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:7:0		**Offset Start:** 0040h at 20h **Offset End:** 0043h at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Version	Enables IEEE1588-2008 support • '0' - IEEE1588 v1 only (bits 20:16 ignored) • '1' - IEEE1588 v1 and IEEE1588-2008 (bit 1 ignored)		0	RW
30 21	RSVD	Reserved for future use. Must be written as '0'.		0	RW
20 16	Mode	Selects timestamping configuration: • '0' - Timestamp PTP Version 1 SYNC and DELAY _REQ messages only. Only IEEE1588 L4 frames are decoded. Timestamps are locked. • '1' - Timestamp all PTP Version 1 messages. When channel is Ethernet, then decode for only IEEE1588 L4 frames. Timestamps are not locked. • '2 - Timestamp PTP Version 1 and 2 event messages only. When channel is Ethernet, then decode for both L4 and L2 IEEE1588 frames. Timestamps are locked. • '3' - Timestamp all PTP Version 1 and 2 messages. When channel is Ethernet, then decode for both L4 and L2 IEEE1588 frames.Timestamps are not locked. • '8' - Timestamp user defined messages. Timestamps are locked. Refer to Table 41-8, "Timestamping Configurations" on page 1633 NOTE: These settings are ignored for the CAN channels		0	RW
15 3	RSVD	Reserved for future use. Must be written as '0'.		0	RW

Table 41-27. Offset 0040h: TS_Ch_Control[0-7] - Time Synchronization Channel Control Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0040h at 20h **Offset End:** 0043h at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
2 : 2	crst	**Channel Reset**. When a '1' is written to this bit, the channel is returned to the same default state as when a power-on reset occurs. After writing a '1' to this bit to reset the logic, the firmware must write a '0' to the bit to indicate the end of the reset.		0h	RW
1 : 1	ta	**Timestamp All messages.** • When this bit is set, the locking of the time snapshot registers is inhibited. Each message is timestamped at the reception of a start of frame delimiter (SFD), regardless of whether the message is a Sync or Delay Request message. The timestamp is captured by the Snapshot register which is never locked and therefore must be read before the next SFD is received. • When this bit is cleared, the timestamp taken after the SFD is frozen or locked when a valid Sync or Delay Request message is detected, until the software resets it.		0h	RW
0 : 0	mm	**Master Mode.** • When this bit is set, it indicates that this channel is a time master on the network. • When cleared, this bit indicates that this channel is in slave mode. The default after reset is slave mode.		0h	RW

41.6.1.18 Offset 0044h: TS_CH_EVENT[0-7] - Time Synchronization Channel Event Register (Per Ethernet Channel)

Register Name	TS_Ch_Event	Access	(See below.)	Reset Value	x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														rxs	txs

*Address offsets per channel...
Channel 0 = 0x044
Channel 1 = 0x064
Channel 2 = 0x084
Channel 3 = 0x0A4
Channel 4 = 0x0C4
Channel 5 = 0x0E4
Channel 6 = 0x104
Channel 7 = 0x124

Table 41-28. Offset 0044h: TS_CH_EVENT[0-7] - Time Synchronization Channel Event Register Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0044h at 20h **Offset End:** 0047h at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 2	Reserved	Reserved for future use.		0h	RV
1 : 1	rxs	**Receive Snapshot Locked.** This bit is automatically set when a Delay_Req message in Master mode, or a Sync message in Slave mode, is received and the **ta** bit in the corresponding TS_Channel_Control register is clear. It indicates that the current system time value has been captured in the RECV_Snapshot register and that further changes to the RECV_Snapshot are now locked out. To clear this bit, write a '1' to it.		0h	RWC
0 : 0	txs	**Transmit Snapshot Locked.** This bit is automatically set when a Sync message in Master mode, or a Delay_Req message in Slave mode, is transmitted and the ta bit in the corresponding TS_Channel_Control register is clear. It indicates that the current system time value has been captured in the XMIT_Snapshot register and that further changes to the XMIT_Snapshot are now locked out. To clear this bit, write a '1' to it.		0h	RWC

41.6.1.19 Offset 0048h: TS_TxSnapLo[0-7] - Transmit Snapshot Low Register (Per Ethernet Channel)

Register Name	TS_TxSnapLo																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XMIT_Snapshot_Low[31:0]																															
*Address offsets per channel... Channel 0 = 0x048 Channel 1 = 0x068 Channel 2 = 0x088 Channel 3 = 0x0A8 Channel 4 = 0x0C8 Channel 5 = 0x0E8 Channel 6 = 0x108 Channel 7 = 0x128																															

Table 41-29. Offset 0048h: TS_TxSnapLo[0-7] - Transmit Snapshot Low Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0048h at 20h **Offset End:** 004Bh at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	XMIT_ Snapshot_ Low	When a Sync message in Master mode, or a Delay_Req message in Slave mode, is transmitted, the current system time is captured in this XMIT_Snapshot register. • The XMIT_Snapshot_Low register contains the lower 32 bits of the time value. • The XMIT_Snapshot_High register contains the upper 32 bits. After a XMIT_Snapshot has occurred, the **txs** indication in the TS_Channel_Event register does not clear until the user writes a '1' to that bit in that register. Therefore, the firmware should read the XMIT_Snapshot_Low and XMIT_Snapshot_High registers before it writes a '1' to the **txs** bit to clear the snapshot indication. In this way, the snapshot value cannot change between reads of the high and low locations.		0000h	RO

41.6.1.20 Offset 004Ch: TS_TxSnapHi[0-7] - Transmit Snapshot High Register (Per Ethernet Channel)

Register Name	TS_TxSnapHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XMIT_Snapshot_High[31:0]																															
*Address offsets per channel... Channel 0 = 0x04C Channel 1 = 0x06C Channel 2 = 0x08C Channel 3 = 0x0AC Channel 4 = 0x0CC Channel 5 = 0x0EC Channel 6 = 0x10C Channel 7 = 0x12C																															

Table 41-30. Offset 004Ch: TS_TxSnapHi[0-7] - Transmit Snapshot High Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 004Ch at 20h **Offset End:** 004Fh at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	XMIT_ Snapshot_ High	When a Sync message in Master mode, or a Delay_Req message in Slave mode, is transmitted, the current system time is captured in this XMIT_Snapshot register. • The XMIT_Snapshot_Low register contains the lower 32 bits of the time value. • The XMIT_Snapshot_High register contains the upper 32 bits. After a XMIT_Snapshot has occurred, the **txs** indication in the TS_Channel_Event register does not clear until the user writes a '1' to that bit in that register. Therefore, the firmware should read the XMIT_Snapshot_Low and XMIT_Snapshot_High registers before it writes a '1' to the **txs** bit to clear the snapshot indication. In this way, the snapshot value cannot change between reads of the high and low locations.		0000h	RO

41.6.1.21 Offset 0050h: TS_RxSnapLo[0-7] - Receive Snapshot Low Register (Per Ethernet Channel)

Register Name	TS_RxSnapLo																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RECV_Snapshot_Low[31:0]																															
*Address offsets per channel... Channel 0 = 0x050 Channel 1 = 0x070 Channel 2 = 0x090 Channel 3 = 0x0B0 Channel 4 = 0x0D0 Channel 5 = 0x0F0 Channel 6 = 0x110 Channel 7 = 0x130																															

Table 41-31. Offset 0050h: TS_RxSnapLo[0-7] - Receive Snapshot Low Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0050h at 20h **Offset End:** 0053h at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	RECV_ Snapshot_ Low	When a Delay_Req message in Master mode, or a Sync message in Slave mode, is received, the current system time is captured in this RECV_Snapshot register. • The RECV_Snapshot_Low register contains the lower 32 bits of the time value. • The RECV_Snapshot_High register contains the upper 32 bits. After a RECV_Snapshot has occurred, the **rxs** indication in the TS_Channel_Event register does not clear until the user writes a '1' to that bit in that register. Therefore, the firmware should read the RECV_Snapshot_Low and RECV_Snapshot_High registers before it writes a '1' to the **rxs** bit to clear the snapshot indication. In this way, the snapshot value cannot change between reads of the high and low locations.		0000h	RO

41.6.1.22 Offset 0054h: TS_RxSnapHi[0-7] - Receive Snapshot High Register (Per Ethernet Channel)

Register Name	TS_RxSnapHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RECV_Snapshot_High[31:0]																															
*Address offsets per channel... Channel 0 = 0x054 Channel 1 = 0x074 Channel 2 = 0x094 Channel 3 = 0x0B4 Channel 4 = 0x0D4 Channel 5 = 0x0F4 Channel 6 = 0x114 Channel 7 = 0x134																															

Table 41-32. Offset 0054h: TS_RxSnapHi[0-7] - Receive Snapshot High Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:7:0		**Offset Start:** 0054h at 20h **Offset End:** 0057h at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	RECV_ Snapshot_ High	When a Delay_Req message in Master mode, or a Sync message in Slave mode, is received, the current system time is captured in this RECV_Snapshot register. • The RECV_Snapshot_Low register contains the lower 32 bits of the time value. • The RECV_Snapshot_High register contains the upper 32 bits. After a RECV_Snapshot has occurred, the **rxs** indication in the TS_Channel_Event register does not clear until the user writes a '1' to that bit in that register. Therefore, the firmware should read the RECV_Snapshot_Low and RECV_Snapshot_High registers before it writes a '1' to the **rxs** bit to clear the snapshot indication. In this way, the snapshot value cannot change between reads of the high and low locations.		0000h	RO

41.6.1.23 Offset 0058h: TS_SrcUUIDLo[0-7] - Source UUID0 Low Register (Per Ethernet Channel)

Register Name	TS_SrcUUID0Lo															Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SourceUUID0_Low[31:0]																															
*Address offsets per channel... Channel 0 = 0x058 Channel 1 = 0x078 Channel 2 = 0x098 Channel 3 = 0x0B8 Channel 4 = 0x0D8 Channel 5 = 0x0F8 Channel 6 = 0x118 Channel 7 = 0x138																															

Table 41-33. Offset 0058h: TS_SrcUUIDLo[0-7] - Source UUID0 Low Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0058h at 20h **Offset End:** 005Bh at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 0	SourceUUID0_Low	When a Delay_Req message in Master mode, or a Sync message in Slave mode, is received with no errors, the Source UUID of the message is captured. The source UUID is located in bytes 64 through 69 of the Ethernet message, and this register contains the lower 32 bits of the source UUID. This register is read-only. At reset, the value in the register is 0, which is not a valid Source UUID value.		0000h	RO

41.6.1.24 Offset 005Ch: TS_SrcUUIDHI[0-7] - SequenceID/SourceUUID High Register (Per Ethernet Channel)

When a Delay_Req message in Master mode, or a Sync message in Slave mode, is received with no errors, the source UUID and the sequence ID of the message are captured.

Register Name	TS_SrcUUIDHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SequenceID[15:0]																SourceUUID_High[47:32]																
*Address offsets per channel... Channel 0 = 0x05C Channel 1 = 0x07C Channel 2 = 0x09C Channel 3 = 0x0BC Channel 4 = 0x0DC Channel 5 = 0x0FC Channel 6 = 0x11C Channel 7 = 0x13C																															

Table 41-34. Offset 005Ch: TS_SrcUUIDHI[0-7] - SequenceID/SourceUUID High Register (Per Ethernet Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 005Ch at 20h **Offset End:** 005Fh at 20h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	SequenceID	The sequence ID is located in bytes 72 and 73 of the Ethernet message, and is captured in this register in bit locations [31:16].		0000h	RO
15 : 0	SourceUUID_High	This register contains the upper 16 bits (bits 47:32) of the source UUID in bit locations [15:0].		0000h	RO
Note: This register has no meaning and is not used for the CAN interfaces.					

41.6.1.25 Offset 0140h: TS_CANx_Status[0-1] - Time Synchronization Channel Event Register (Per CAN Channel)

Register Name	TS_CANx_Status																Access				(See below.)				Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														valid	ovr
*Address offsets per channel... CAN 0 = 0x140 CAN 1 = 0x150																															

Table 41-35. Offset 0140h: TS_CANx_Status[0-1] - Time Synchronization Channel Event Register (Per CAN Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0140h at 10h **Offset End:** 0143h at 10h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	Reserved	Reserved for future use.		0h	RV
1 : 1	valid	**Snapshot Valid.** This bit is automatically set when a CAN interrupt has caused a snapshot to be taken. It indicates that the current system time value has been captured in the `CAN_Snapshot` register t. This bit remains set until the firmware writes a '1' to this bit location.		0h	RWC
0 : 0	ovr	**Snapshot Overrun.** If a second snapshot is taken while the valid flag is still set, then the overrun error bit (ovr) in this register is set to a '1'. This indication notifies the firmware that a previous snapshot was overwritten by the current snapshot and never read. To clear this bit, write a '1' to it.		0h	RWC

41.6.1.26 Offset 0144h: TS_CANSnapLo[0-1] - Transmit Snapshot Low Register (Per CAN Channel)

Register Name	TS_CANSnapLo																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAN_Snapshot_Low[31:0]																															
*Address offsets per channel... Channel 0 = 0x144 Channel 1 = 0x154																															

Table 41-36. Offset 0144h: TS_CANSnapLo[0-1] - Transmit Snapshot Low Register (Per CAN Channel)

Description:					
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:7:0		**Offset Start:** 0144h at 10h **Offset End:** 0147h at 10h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CAN_ Snapshot_ Low	When a CAN packet is transmitted or received, the current system time is captured in this CAN_Snapshot register. • The CAN_Snapshot_Low register contains the lower 32 bits of the time value. • The CAN_Snapshot_High register contains the upper 32 bits. After a CAN_Snapshot has occurred, the **valid** indication in the TS_CAN_Status register does not clear until the user writes a '1' to that bit in that register. The firmware should check the state of the valid bit in the CAN_Status register before reading CAN_Snapshot_Low. Because the snapshot value could change between reads of the Low and High snapshot registers, the firmware should check the state of the overrun bit before and after the read of the CAN_Snapshot_High register. After reading the CAN_Snapshot_High register, the firmware should write a '1' to the valid bit and the overrun bit if applicable.		0000h	RO

41.6.1.27 Offset 0148h: TS_CANSnapHi[0-1] - Transmit Snapshot High Register (Per CAN Channel)

Register Name	TS_CANSnapHi																Access				(See below.)				Reset Value				0x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAN_Snapshot_High[31:0]																															
*Address offsets per channel... CAN Channel 0 = 0x148 CAN Channel 1 = 0x158																															

Table 41-37. Offset 0148h: TS_CANSnapHi[0-1] - Transmit Snapshot High Register (Per CAN Channel)

Description:					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:7:0	**Offset Start:** 0148h at 10h **Offset End:** 014Bh at 10h	
Size: 32 bits	**Default:** 0000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 00	CAN_ Snapshot_ High	When a CAN packet is transmitted or received, the current system time is captured in this CAN_Snapshot register. • The CAN_Snapshot_Low register contains the lower 32 bits of the time value. • The CAN_Snapshot_High register contains the upper 32 bits. After a CAN_Snapshot has occurred, the **valid** indication in the TS_CAN_Status register does not clear until the user writes a '1' to that bit in that register. The firmware should check the state of the valid bit in the CAN_Status register before reading CAN_Snapshot_Low. Because the snapshot value could change between reads of the Low and High snapshot registers, the firmware should check the state of the overrun bit before and after the read of the CAN_Snapshot_High register. After reading the CAN_Snapshot_High register, the firmware should write a '1' to the valid bit and the overrun bit if applicable.		0000h	RO

41.6.1.28 Offset 01F0h: TS_Aux_TrgtLo - Auxiliary Target Time Low Register

Register Name	TS_Aux_TrgtLo																Access	(See below.)							Reset Value	0x0000_0000							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AuxTargetTime_Low[31:0]																																

Table 41-38. Offset 01F0h: TS_Aux_TrgtLo Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 000001F0h Offset End: 000001F3h	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 0	Aux_TargetTime_Low	The Auxiliary Target Time register set contains 64 bits of a time value. When the system time is greater than or equal to the auxiliary target time value, an interrupt is generated to the Host on the **ts_intreq** signal if the **atm** bit in the Time Sync Control register is set. For more information about the Auxiliary Target Time interrupt, see Section 102.8.2.1, "Time Sync Control Register" on page 4775.		0000h	RW

41.6.1.29 Offset 01F4h: TS_Aux_TrgtHi -Auxiliary Target Time High Register

Register Name	TS_Aux_TrgtHi																Access	(See below.)							Reset Value	0x0000_0000							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TargetTime_Low[31:0]																																

Table 41-39. Offset 01F4h: TS_Aux_TrgtHi Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 000001F4h Offset End: 000001F7h	
Size: 32 bits	Default: 0000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 0	Aux_TargetTime_High	The Auxiliary Target Time register set contains 64 bits of a time value. When the system time is greater than or equal to the auxiliary target time value, an interrupt is generated to the Host on the **ts_intreq** signal if the **atm** bit in the Time Sync Control register is set. For more information about the Auxiliary Target Time interrupt, see Section 102.8.2.1, "Time Sync Control Register" on page 4775.		0000h	RW

41.6.1.30 Offset 0200h: L2_EtherType - L2 EtherType Register

Register Name	L2_EtherType																Access				(See below.)				Reset Value				x0000_88F7			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)																EtherType															

02

Table 41-40. Offset 0200h: L2 EtherType Register

Description:				
View: PCI	BAR: CSRBAR	Bus:Device:Function: M:7:0		Offset Start: 00000200h Offset End: 00000203h
Size: 32 bits	Default: 000088F7h			Power Well: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	RSVD	Reserved for future use. Must be written as '0'		0	RO
15 : 0	EtherType	Ethertype compare value (default = 0x88F7). The user may optionally supply a different Ethertype compare value for L2 IEEE1588 detection		88F7h	RW

41.6.1.31 Offset 0204h: UD_EtherType - User Defined EtherType Register

Register Name	UD_EtherType																Access				(See below.)				Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mask																UD_EtherType															

Table 41-41. Offset 0204h: User Defined EtherType Register

Description:				
View: PCI	BAR: CSRBAR	Bus:Device:Function: M:7:0		Offset Start: 0000204h Offset End: 0000207h
Size: 32 bits	Default: 00000000h			Power Well: Core
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 16	Mask	Mask for compare value		0h	RW
15 : 0	UD_EtherType	User defined compare value for ethertype field. Used in conjunction with mask, above.		0h	RW

41.6.1.32 Offset 0208h: UD_Header_Offset - User Defined Header Offset Register

Register Name	UD_Header_Offset																Access				(See below.)				Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)																UD_Offset															

Table 41-42. Offset 0208h:User Defined Header Offset Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 00000208h Offset End: 0000020Bh	
Size: 32 bits	Default: 00000000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 8	RSVD	Reserved. Must be written as “0”		0h	RW
7 : 0	UD_Offset	User defined offset for header		0h	RW

41.6.1.33 Offset 020Ch: UD_Header - User Defined Header Register

Register Name	UD_Header																Access				(See below.)				Reset Value				x0000_0000			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mask																UD_Header															

Table 41-43. Offset 020Ch:User Defined Header Register

Description:					
View: PCI	BAR: CSRBAR		Bus:Device:Function: M:7:0	Offset Start: 0000020Ch Offset End: 0000020Fh	
Size: 32 bits	Default: 00000000h			Power Well: Core	
Bit Range	Bit Acronym	Bit Description	Sticky	Bit Reset Value	Bit Access
31 : 16	Mask	Mask for compare value		0h	RW
15 : 0	UD_Header	User defined compare value for header		0h	RW

§ §

42.0 Local Expansion Bus Controller

42.1 Overview

The Expansion bus controller provides an interface from to external Expansion target devices.

The Expansion bus controller includes a 25-bit address bus and a 16-bit wide data path. The expansion bus supports Intel multiplexed, Intel non-multiplexed, Intel StrataFlash® technology, Intel StrataFlash® Synchronous Memory, Micron* Flow-Through ZBT, Motorola* multiplexed, Motorola non-multiplexed, and Texas Instruments* Host Port Interface (HPI) target devices.

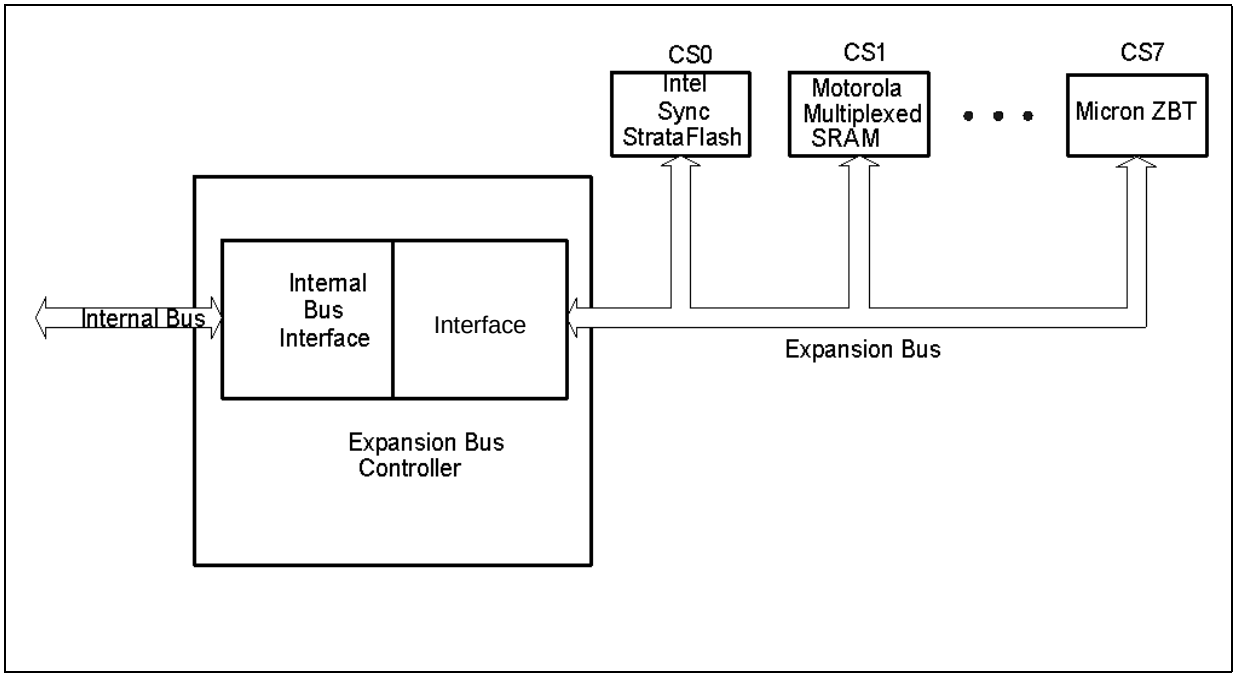
Byte-wide devices may also be used by programming the channel connection for 8-bit operation. For TI DSPs that support an internal bus width of 32 bits, the multiplexed HPI-8 or HPI-16 interface can be used to complete these transfers.

42.2 Feature List

- Outbound transfers (the EP80579 integrated processor is the master to an external target device).
- Eight programmable target chip selects.
- Twenty five bits of address; sixteen bits of data.
- Supports Intel mode and Motorola* mode bus cycles.
- Supports Intel StrataFlash® technology.
- Supports 66-MHz Intel StrataFlash® Synchronous Memory (16-bit only).
- Supports 16-bit Micron Flow-Through ZBT (Zero bus turnaround) SRAMS.
- Supports 8-bit and 16-bit Texas Instruments HPI specifications.
- Multiplexed or non-multiplexed address / data buses for Intel/Motorola*/HPI bus cycles.
- Supports even and odd parity generation and calculation for Intel/Motorola*/ Micron* ZBT modes.
- Maximum clock input frequency of 80 MHz.
- Minimum input clock frequency of 33MHz.

42.3 Block Diagram

Figure 42-1. Expansion Bus Controller



42.4 Theory of Operation

The Expansion bus controller supports outbound transfers that are initiated by the EP80579 that target Expansion bus slaves. The Expansion data bus is 16 bits wide and the address bus is 25 bits wide.

Since the Expansion bus controller has only 1 outbound transaction queue, outbound accesses all complete in order.

42.4.1 Outbound Transfers

For outbound data transfers, the Expansion bus controller occupies up to 256 Mbytes of address space in the EP80579's memory map (refer to signal EX_ADDR [24:0] in Table 48-24, "Expansion Bus Signals") and contains a 1-deep address queue, an 8-word write data fifo, and an 8-word read data FIFO. Eight chip selects are supported to allow up to eight independent external devices to be connected. The address space for each chip select is up to 32 Mbytes.

An external clock input, EX_CLK, is required to operate the Expansion interface. The maximum clock frequency supported by the Expansion bus controller is 80 MHz. The clock input is provided to allow a wide variety of different peripherals to be connected to the Expansion interface.To provide a glue-less interface to a wide variety of devices, the Expansion bus controller supplies eight chips selects to a 16-bit wide external bus, which can be configured as Intel, Synchronous Intel, Micron ZBT, Motorola, or HPI-style controls. The signaling characteristics and timing for each chip select is individually programmable.

For Synchronous Intel StrataFlash Memory, the Expansion bus controller only supports single word asynchronous page-mode read and synchronous burst-mode read (1-8 words). It does not support page mode read mode or single word latched asynchronous read mode. When configuring a Synchronous Intel StrataFlash Memory, wait polarity must be programmed to active low, data hold programmed to one clock, wait delay be deasserted with valid data and clock edge programmed to rising edge. For 16-bit Synchronous Intel devices, the burst length must be programmed to 16-word bursts. The latency count must be programmed to the appropriate value that is defined in the specific Synchronous Intel StrataFlash Memory specification.

The Expansion Bus interface signals need to be connected based upon the device type (Intel, Synchronous Intel, Micron ZBT, Motorola, or HPI-style control signals) and a sample mapping of the pins are shown in Table 42-1.

Table 42-1. Example Expansion Bus Pin Mappings to Target Devices

Pin	Intel StrataFlash® 28F128J3A	Synchronous Intel StrataFlash 28F256K3	Motorola* MCM6946	Micron* ZBT MT55L128L36F1	TI* HPI TMS320UC5409
EX_ALE	OPEN	ADV#	OPEN	ADV/LD#	OPEN
EX_ADDR[24:0]	A[23:0]	A[24:1]	A[18:0]	SA	HCSEL,HCNTL,HBIL
EX_BE_N[1:0]	OPEN	OPEN	OPEN	BW[d:a]#	OPEN
EX_CS_N[7:0]	CE#	CE#	EN	CE#	hcs
EX_DATA[15:0]	D[15:0]	D[15:0]	DQ[7:0]	DQ	HD[7:0]
EX_IOWAIT_N	OPEN	OPEN	OPEN	OPEN	OPEN
EX_PARITY[1:0]	OPEN	OPEN	OPEN	DQ	OPEN
EX_RD_N	OE#	OE#	G_N	OE#	hr_w_n
EX_RDY_N[3:0]	OPEN	OPEN	OPEN	OPEN	hrdy
EX_WR_N	WE#	WE#	W_N	R/W#	hds1_n

The EX_IOWAIT_N signal is available to be shared by the devices attached to chip 0 through 7, when the chip selects are configured in Intel or Motorola mode of operation. The EX_IOWAIT_N signal allows an external device to hold off completion of the read or write phase of a transaction until the external device is ready to complete the transaction.

Similarly, EX_RDY[3:0] are provided for chip selects 7 through 4, respectively. The EX_RDY[3:0] signals are used to hold off data transfers when chip selects 7 through 4 are configured in HPI mode. For example when chip select 5 is configured in HPI mode of operation, chip select 5 will no longer respond to the EX_IOWAIT_N signal and will only respond to the EX_RDY_N[1]. All other chip selects will respond to the EX_IOWAIT_N signal. Chip selects 7 through 4 are the only chip selects that can be configured in HPI mode of operation.

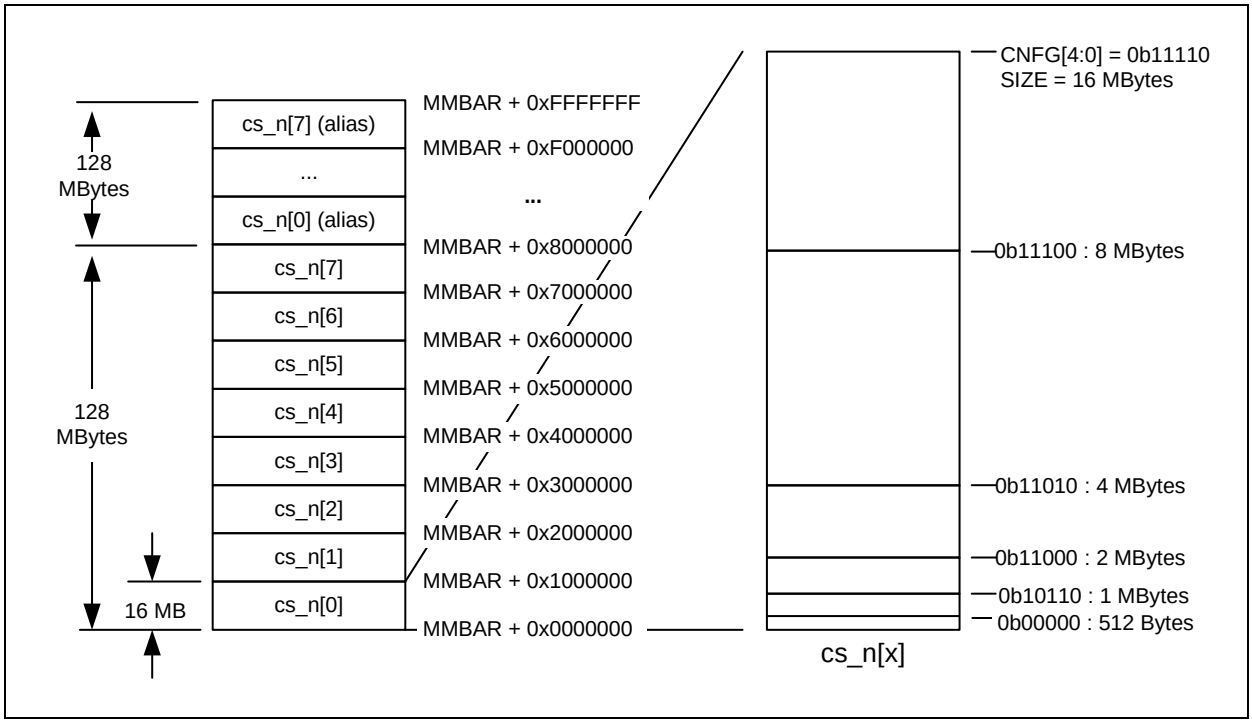
42.4.1.1 Chip Select Address Allocation

The Expansion bus controller occupies up to 256 Mbytes of address space in the EP80579 memory map. The Expansion bus controller uses bits 27:0, from the internal bus, to determine how to translate the internal bus address to the Expansion Bus Address. If there are no 32 MByte devices programmed (i.e., All eight EXP_TIMING_CS registers bit 9 equal 0), the lower 24 bits of the internal bus address are translated to the lower 24 bits of the Expansion Bus address, EX_ADDR [23:0]. EX_ADDR[24] will always be zero. Bits 26:24 of the internal bus are used to decode one of eight chip-

select regions implemented by the Expansion bus, each region being 16 Mbytes. Address bit 27 is not used and will currently alias each chip select region as shown on the left side of Figure 42-2.

The right side of Figure 42-2 shows the implementation of bit 13:10 of the each Timing

Figure 42-2. Chip Select Address Allocation When There Are no 32-MByte Devices Programmed



and Control (EXP_TIMING_CS) Register. A Timing and Control (EXP_TIMING_CS) Register is implemented for each of the eight chip selects. Each chip select defines a base region size of 512 bytes with the actual size of the region given by the formula shown in Figure 42-3. If the address is outside of the programmed region, the Expansion bus controller responds with an error.

Figure 42-3. Expansion Bus Memory Sizing

Region Size = $2^{(9+CNFG[4:1] + 16*CNFG[0])}$

For Examples of how to use this feature:

If bits 13:9 of Timing and Control (EXP_TIMING_CS0) Register 0 = “00000” an address space of 2^9 = 512 Bytes is defined for chip select 0 (EX_CS0_N).

If bits 13:9 of Timing and Control (EXP_TIMING_CS1) Register 1 = “10000” an address space of 2^{17} = 128KBytes is defined for chip select 1 (EX_CS1_N).

If bits 13:9 of Timing and Control (EXP_TIMING_CS2) Register 2 = “11110” an address space of 2^{24} = 16Mbytes is defined for chip select 2 (EX_CS2_N).

If bits 13:9 of Timing and Control (EXP_TIMING_CS7) Register 7 = “00001” an address space of 2^{25} = 32Mbytes is defined for chip select 7 (EX_CS7_N).

If there is a 32-MByte device programmed in any of the eight EXP_CS_TIMING registers, a different memory map is used as shown in Figure 42-4. The lower 25 bits of the internal bus address are translated to the lower 25 bits of the Expansion Bus address, EX_ADDR [24:0]. Bits 27:25 of the internal bus are used to decode one of eight chip-select regions implemented by the Expansion bus, each region being 32 Mbyte. If a design has 16-MByte or smaller devices on all of the chip selects, one of the EXP_CS_TIMING register could be programmed to a 32-Mbyte device so the Expansion bus address mapping will not change if that design switches to 32-MByte device sometime in the future. The Expansion bus controller will still work with the smaller device, however an error response will not be generated if there is an access outside the device window for that device.

42.4.1.2 Address and Data Byte Steering

Figure 42-4. Chip Select Address Allocation when a 32 Mbyte device is programmed

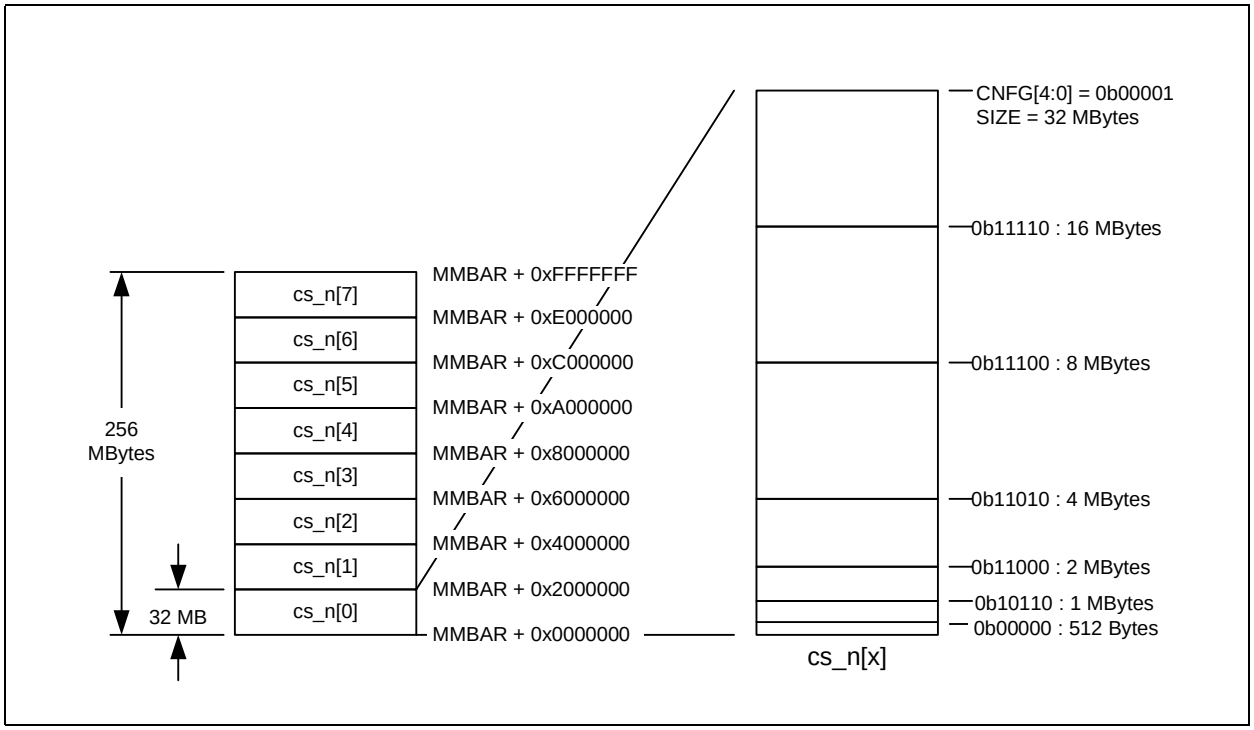


Table 42-2 shows the address and data mapping from the internal bus to the Expansion Bus. This table applies to Intel, Synchronous Intel, Micron ZBT and Motorola defined cycles only. For 32-bit read operations to a byte/halfword wide interface, multiple bytes are collected and then transferred as a complete 32-bit word. This pattern occurs as shown below for any allowable sub-length read access. Four and eight word reads are also supported and generate multiple accesses to the target device. Four and eight word reads to Synchronous Intel, only generate one burst access to the device. Byte enables are generated for both reads and writes and are valid the same cycles (T1-T4 phases) as EX_ADDR is valid. Byte write devices (devices that need EX_BE_N asserted in the same exact cycles that EX_WR_N is asserted) are not supported. The internal bus is always big endian format; the Expansion bus is big endian as well. The data byte steering for each cycle type is showing in Table 42-2.

For 8-bit devices, EX_DATA[31:8] must not toggle to conserve power. Similarly, for 16-bit devices, EX_DATA[31:16] must not toggle. For sub-word writes to 32-bit devices, EX_DATA must not toggle for byte enables not asserted.

42.4.1.3 Expansion Bus Interface Configuration

Table 42-2. Expansion Bus Address and Data Byte Steering (Sheet 1 of 2)

Internal Bus Cycle	Device Width Connected to Expansion Bus (8-bit or 16-bit)	Internal Address Value (Internal _ADDR[1:0])	Expansion Bus Address Value (EX_ADDR[1:0])	Data Location Translation Between Expansion Data Bus and Internal Data Bus
Byte write	8-bit	00	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
			01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
			10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
			11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word write	16-bit	00	00	Internal data bus [31:16] = Expansion data bus [15:0], EX_BE_N = 0x0
			10	Internal data bus [15:0] = Expansion data bus [15:0], EX_BE_N = 0x0
Byte read	8-bit	00	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
			01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
			10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
			11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word read	16-bit	00	00	Internal data bus [31:16] = Expansion data bus [15:0], EX_BE_N = 0x0
			10	Internal data bus [15:0] = Expansion data bus [15:0], X_BE_N = 0x0
Byte read	8-bit	0x	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
			01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte read	8-bit	1x	10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
			11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word read	16-bit	0x	00	Internal data bus [31:16] = Expansion data bus [15:0], EX_BE_N = 0x0
Word read	16-bit	1x	10	Internal data bus [15:0] = Expansion data bus [15:0], EX_BE_N = 0x0
Byte write	8-bit	0x	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
			01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte write	8-bit	1x	10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
			11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word write	16-bit	0x	00	Internal data bus [31:16] = Expansion data bus [15:0], EX_BE_N = 0x0

Table 42-2. Expansion Bus Address and Data Byte Steering (Sheet 2 of 2)

Internal Bus Cycle	Device Width Connected to Expansion Bus (8-bit or 16-bit)	Internal Address Value (Internal _ADDR[1:0])	Expansion Bus Address Value (EX_ADDR[1:0])	Data Location Translation Between Expansion Data Bus and Internal Data Bus
Word write	16-bit	1x	10	Internal data bus [15:0] = Expansion data bus [15:0], EX_BE_N = 0x0
Byte read	8-bit	00	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte read	8-bit	01	01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte read	8-bit	10	10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte read	8-bit	11	11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word read	16-bit	00	00	Internal data bus [31:24] = Expansion data bus [15:8], EX_BE_N = 0x1
Word read	16-bit	01	00	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Word read	16-bit	10	10	Internal data bus [15:8] = Expansion data bus [15:8], EX_BE_N = 0x1
Word read	16-bit	11	10	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte write	8-bit	00	00	Internal data bus [31:24] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte write	8-bit	01	01	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte write	8-bit	10	10	Internal data bus [15:8] = Expansion data bus [7:0], EX_BE_N = 0x2
Byte write	8-bit	11	11	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2
Word write	16-bit	00	00	Internal data bus [31:24] = Expansion data bus [15:8], EX_BE_N = 0x1
Wordwrite	16-bit	01	00	Internal data bus [23:16] = Expansion data bus [7:0], EX_BE_N = 0x2
Word write	16-bit	10	10	Internal data bus [15:8] = Expansion data bus [15:8], EX_BE_N = 0x1
Word write	16-bit	11	10	Internal data bus [7:0] = Expansion data bus [7:0], EX_BE_N = 0x2

There are eight registers called the Timing and Control (EXP_TIMING_CS) Registers that define the operating mode for each chip select. When designing with the Expansion Bus Interface, placing the devices on the correct chip selects is required.

Chip Select 0 through 7 can be configured to operate with devices that require an Intel, Synchronous Intel, Micron ZBT or Motorola Micro-Processor style bus accesses. These chip selects can be configured to operate in a multiplexed or a simplex mode of operation for either Intel- or Motorola-style bus accesses. Additionally, Chip Select 4 through 7 can be configured to generate Texas Instruments HPI-style bus accesses. The mode of operation (Intel, Motorola, TI HPI, or Synchronous Intel, Micron ZBT) is set by bits 15,14, and 8 of each Timing and Control (EXP_TIMING_CS) Register. Table 42-6 on page 1698 shows the possible settings for the Cycle Type selection using bits 15, 14, and 8 of the Timing and Control (EXP_TIMING_CS) Register.

Once the cycle type has been determined, the mode of operation must be set. There are two configurable modes of operation for each chip select, multiplexed and non-multiplexed. Bit 4 of the Timing and Control (EXP_TIMING_CS) Registers is used to select this mode. If bit 4 of the Timing and Control (EXP_TIMING_CS) Register is set to logic 1, the access mode for that Chip Select is multiplexed. Likewise, if bit 4 of the Timing and Control (EXP_TIMING_CS) Register is cleared to logic 0, the access mode for that Chip Select is non-multiplexed. For Synchronous Intel, Micron ZBT memories bit 4 must be programmed to logic 0. Multiplexed and non-multiplexed can imply different operations depending upon the Cycle Type that is selected.

The size of the data bus for each device connected to the Expansion bus must be configured. The data bus size is selected on a per-chip-select basis, allowing the most flexibility when connecting devices to the Expansion bus. There are two valid selections that can be configured for each data bus size, 8-bit or 16-bit. Bit 0 f each Timing and Control (EXP_TIMING_CS) Register is used to select the data bus size on a per-chip-select basis.

Each chip select can be independently enabled or disabled by setting a value in bit 31 of each Timing and Control (EXP_TIMING_CS) Register. Clearing bit 31 of the Timing and Control (EXP_TIMING_CS) Register to logic 0 disables the corresponding chip select. Setting bit 31 of the Timing and Control (EXP_TIMING_CS) Register to logic 1 enables the corresponding chip select. Accesses to chip selects that are disabled result in an error response.

Split transfers are supported for all read transfer types and controlled by setting bit 3 (SPLT_EN) of the Timing and Control (EXP_TIMING_CS) Register. Setting bit 3 of each Timing and Control (EXP_TIMING_CS) Register to logic 1 enables split transfers for accesses to the corresponding chip select. Clearing bit 3 of each Timing and Control (EXP_TIMING_CS) Register to logic 0 disables split transfers for accesses to the corresponding chip select. Enabling split transactions allows for more efficient utilization of the internal bus, especially for slow external expansion bus devices. For higher performance devices with low read latencies, disabling split transactions may provide better performance.

Each chip select region has the ability to be write-protected by setting bit 1 of each Timing and Control (EXP_TIMING_CS) Register. When bit 1 of Timing and Control (EXP_TIMING_CS) Register is cleared to logic 0, writes to a specified chip select region results in an error response. When bit 1 of Timing and Control (EXP_TIMING_CS) Register is set to logic 1, writes are allowed to a specified chip select region. Chip select 0 will be write-protected after reset.

For chip selects 4 through 7 configured in HPI mode of operation, there is an associated ready bit (EX_RDY [3:0]). The ready bit is only used when the mode of operation is set to Texas Instruments HPI mode. The ready bits are used to hold off the host processor when the given DSP is not ready to complete the transfer. However, the polarity of this ready bit can vary based upon the DSP that is selected. Bit 5 of each Timing and Control (EXP_TIMING_CS) Register allows the polarity used by each ready bit to be independently set. When bit 5 of the Timing and Control (EXP_TIMING_CS) Register is cleared to logic 0, the ready bit is cleared to respond to an active low signal (logic 0). When bit 5 of the Timing and Control (EXP_TIMING_CS) Register is set to logic 1, the ready bit is set to respond to an active high signal (logic 1).

One final set of parameters that may be set prior to using Expansion Bus Interface Chip Select 1 through Chip Select 8. After boot up, these parameters may be adjusted for Chip Select 0 as well. These five parameters are the timing extension parameters for each phase of an Expansion Bus access.

There are five phases to every Expansion Bus access:

- T1 – Address Timing
- T2 – Setup/Chip Select Timing

- T3 – Strobe Timing
- T4 – Hold Timing
- T5 – Recovery Phase

For Synchronous Intel mode, the T1,T2, T3, T4, T5 timing parameters are only used for writes. For Synchronous Intel reads, the Expansion bus controller uses the Count value programmed in the EXP_SYNCINTEL_COUNT register to determine how many cycles before data is valid (based on the timing parameters for a specific device) For Micron ZBT devices, the timing parameters must be programmed to all zero since reads and writes are synchronous.

The Expansion-bus address is used to present the 25 bits of the address [24:0] used for the Expansion bus access accompanied by an address latch enable output signal, EX_ALE for multiplexed devices. The address phase normally lasts two clock cycles in multiplexed mode. The address phase may be extended by one to three clock cycles using the T1 - Address Timing parameter, bits 29:28 in the Timing and Control (EXP_TIMING_CS) Register for the particular Chip Select. When the address phase T1 is extended, the ALE pulse is extended and always deasserts one cycle prior to the end of the T1 phase. The lower address bits are placed onto the data bus (i.e for a 16 bit data bus, EX_DATA contains EX_ADDR[15:0]) along with EX_ADDR[24:0] signals during the first cycle of the address phase. During the second cycle of the address phase, the data bus now will output data when attempting to complete a write or tri-state when attempting to complete a read. The address signals will retain their state.

For Synchronous Intel and Micron ZBT devices, EX_ALE acts as the address valid signal (ADV#) and is logic 0 during the address phase and logic 1 during the continuation of a burst or IDLE cycle.

Due to the fact that, in HPI mode of operation, it is possible to begin an access to a busy device (EX_RDY is false), special consideration must be taken with programming the T1 — Address Timing parameter when using the chip select in HPI mode. The T1 – Address Timing parameter must be set to a minimum of two additional cycles (T1 must equal to 0x2). Programming the T1 – Address Timing parameter to this value ensures that the asynchronous EX_RDY input is sampled and available to the controlling hardware logic before beginning the new HPI access over the Expansion bus.

The chip-select signal is presented for one Expansion bus phase before the Strobe Phase. The chip select will be presented for the remainder of the Expansion bus cycles (setup, strobe, and hold phases).

The Setup/Chip Select Timing phase may also be extended by one to three clock cycles, using bits 27:26 of the Timing and Control (EXP_TIMING_CS) Register, T2 – Setup/Chip Select Timing parameter. In HPI mode of operation, T2 is defined as the time required by the external DSP device to drive EX_RDY false for the current access plus the time required by the Expansion bus controller to sample and synchronize the EX_RDY signal. The T2 – Setup/Chip Select Timing parameter must have a minimum value of two additional cycles (T2 >= 0x2). Programming the T2 – Setup/Chip Select Timing parameter to be three clock cycles in length ensures that when the Strobe Phase, T3, begins, the Strobe Phase will be able to sample the EX_RDY signal and exit the Strobe Phase at the proper time.

The Strobe Phase of an Expansion-bus access is when the read or write strobe is applied. The 25 Expansion Bus Interface Address bits are maintained in non-multiplexed mode or the Expansion Bus Interface Data bus is switched from address to data when configured in multiplexed mode during the Strobe Phase.

The Strobe Phase may be extended from one to 15 clock cycles, as defined by programming bits 25:22 of the Timing and Control (EXP_TIMING_CS) Register, T3 – Strobe Timing parameter. In HPI mode of operation, the T3 – Strobe Timing parameter

must have a minimum value of one additional cycle (T3 >= 0x1). Programming the T3 – Strobe Timing parameter to be two clock cycles in length ensures that any data sent to the DSP is captured regardless of when the EX_RDY signal is asserted by the DSP.

The Hold Phase of an Expansion-bus access is provided to allow a hold time for data to remain valid after the data strobe has transitioned to an invalid state. During a write access, the Hold Phase provides hold time for data written to an external device on the Expansion bus, after the strobe pulse has completed.

During a read access, the Hold Phase allows an external device time to release the bus after driving data back to the controller. The Hold Phase may be extended one to three clock cycles, using bits 21:20 of the Timing and Control (EXP_TIMING_CS) Register, T4 – Hold Timing parameter. In HPI mode of operation, the Hold Phase is defined the same as described for the Intel and Motorola modes of operation, but must be set to a minimum value of one additional cycle (T4 >= 0x1).

After the address and chip select is de-asserted, the Expansion bus controller can be programmed to wait a number of clocks before starting the next Expansion Bus access. This action is referred to as the Recovery Phase. The Recovery Phase is may be extended one to 15 clock cycles using bits 19:16 of the Timing and Control (EXP_TIMING_CS) Register, T5 – Recovery Timing parameter. In HPI mode of operation, the Recovery Phase is defined the same as described for the Intel and Motorola modes of operation.

42.4.1.4 Using I/O Wait

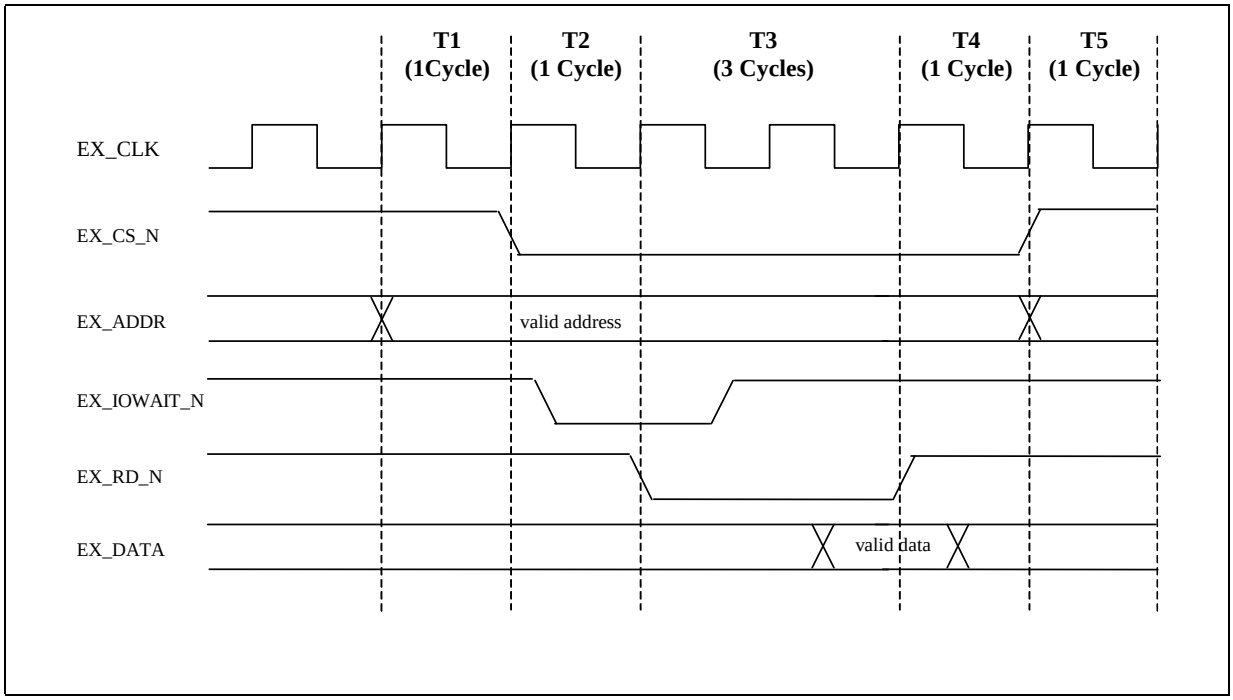
The EX_IOWAIT_N signal is available to be shared by devices attached to chip selects 0 through chip select 7, when configured in Intel or Motorola modes of operation. The shared device will assert EX_IOWAIT_N in the T2 phase of a read or write transaction. During idle cycles, the board is responsible for ensuring that EX_IOWAIT_N is pulled-up. Additionally, EX_IOWAIT_N must always be pulled high during Micron ZBT, Intel Synchronous Mode, and HPI cycles. The Expansion bus controller will ignore EX_IOWAIT_N for Synchronous Intel mode transfers and use the EXP_SYNCINTEL_COUNT register for the wait state generation.

When an external device asserts EX_IOWAIT_N before the first cycle of a Strobe phase of a read or write transaction, the Expansion bus controller will hold in the Strobe phase until the EX_IOWAIT_N signal returns to an inactive state. Since there is a synchronizer cell on EX_IOWAIT_N, the external device must assert EX_IOWAIT_N three cycles before the deassertion of EX_WR_N/EX_RD_N. This implies that the value programmed in the T2 and T3 phase cannot both be equal to zero. After EX_IOWAIT_N is deasserted the Expansion bus controller will only transition to the T4 - Hold state after the T3 counter reaches zero.

Operation of EX_RDY signals is the same as the EX_IOWAIT_N signal, but is defined primarily for the ‘C54xx family of DSPs. In addition, the EX_RDY signals will hold the current access in the Address Phase if detected during that phase. This event can happen if a busy DSP memory access is started before the previous access completes. Figure 42-5 shows the operation of the EX_IOWAIT_N signal.

Notice that the access is an Intel Style Simplex Read access. The data strobe phase is set to a value to last three clock cycles. The data is returned from the peripheral device prior to the three clocks and the peripheral device de-asserts EX_IOWAIT_N. The data strobe phase terminates after two clocks even though the strobe phase was configured to pulse for three clocks.

Figure 42-5. Expansion Bus I/O Wait Operation



42.4.1.5 Parity

The Expansion bus controller generates even or odd parity for each byte written to EX_DATA and compares parity for each byte read on EX_DATA, if PAR_EN is set in each EXP_TIMING_CS register. EX_PARITY is transferred in the same clock cycle in which EX_DATA is transferred. If a read transfer results in a parity mismatch on EX_PARITY, the target address is logged in the EXP_PARITY_STATUS and OutErrorSts is set. Exp_parity_error will be asserted by the Expansion bus controller during a parity error and an interrupt will be generated if enabled in the interrupt controller. Exp_parity_error will remain asserted until software clears EXP_PARITY_STATUS register.

Even parity is defined as the number of 1's on EX_DATA[7:0] and EX_PARITY[0] must be an even number. For example, if EX_DATA[7:0] = 0x25, EX_PARITY[0] must be 1, since there are 3 bits set on 0x25 and there needs to be an even number of 1s. Parity for the second byte of EX_DATA is generated on EX_PARITY[1].

If PAR_EN is cleared for a particular device, the Expansion bus controller doesn't generate or compare parity and EX_PARITY should not toggle to conserve power. Odd parity can be enabled by setting OddPar in the EXP_MST_CONTROL register and is equivalent to the inverted value of even parity.

42.4.1.6 Special Design Knowledge for Using HPI mode

The Expansion bus controller supports a number of the 8-bit and 16-bit versions of the Texas Instruments Host Port Interface* (HPI) standards. This flexibility allows the TMS320C54xx family of Digital Signals Processors (DSP) to seamlessly interface to the Expansion Bus.

However, there are some special things to note when using the Expansion Bus in HPI mode of operation. These features are shown in the following tables. There are also some restrictions on the timing parameters and these are outlined in Section 42.4.1.3, “Expansion Bus Interface Configuration” on page 1676.

The Expansion-bus address-pins bits 0, 1, 2, 22, and 23 are multiplexed with special function signal pins for HPI as shown in Table 42-3.

Table 42-3. Multiplexed Output Pins for HPI Operation

HPI Control Signal	Output Signal Pin
EX_HBIL	EX_ADDR [0]
EX_HCNTL [1:0]	EX_ADDR [2:1]
EX_HCSEL [1:0]	EX_ADDR [23:22]

The byte identification signal, EX_HBIL, is used to determine the byte transfer order. (EX_HBIL is driven low for the first byte of the transfer and driven high for the second byte.)

The byte order bit (BOB) in the HPIC register (contained in the DSP) — within the HPI device — is used to determine the placement for the two bytes of the transfer. Please consult the datasheet of the specific DSP being connected to determine the order of the transferred bytes.

When operating in HPI mode, bits 13:10 in the Timing and Control (EXP_TIMING_CS) Registers are ignored.

When operating in HPI-16, non-multiplexed mode, the Expansion bus address bus provides direct accesses to the DSP memory space. The data associated with this address will be read or written from the location specified by the value contained on the Expansion Bus address bits.The signals EX_HCNTL [1:0] are multiplexed onto the EX_ADDR [2:1] pins. When communicating to a multiplexed HPI interface, the EX_HCNTL [1:0] signals are used to select one of four internal registers used for interfacing to the DSP. The EX_HCNTL [1:0] mapping is described in the Table 42-4.

Table 42-4. HPI HCNTL Control Signal Decoding

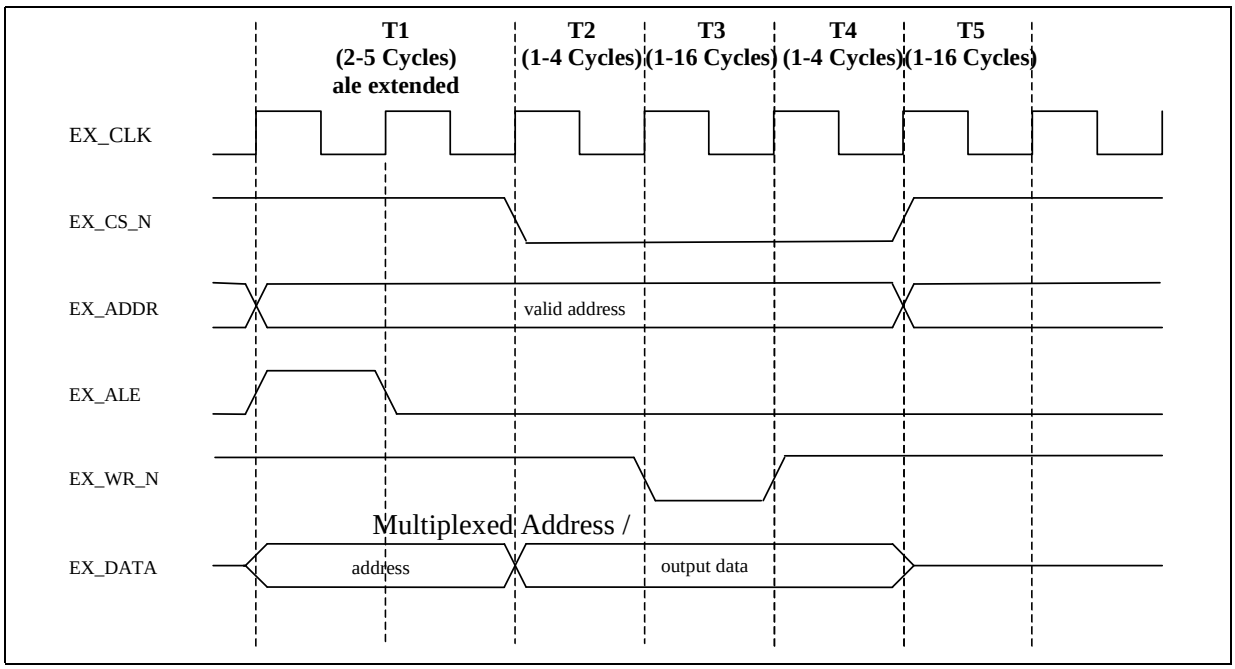
hcntl[1:0]	Required Access
00	Read / write control register (HPIC)
01	Read / write data register (HPID) HPI-8: Post-increment HPIA on reads, pre-increment on writes. HPI-16: Post-increment HPIA on reads and writes
10	Read / write address register (HPIA)
11	Read / write data register (HPID)

42.4.1.7 Expansion Bus Outbound Timing Diagrams

The STATE signal that is shown in some of the following timing diagrams is the internal state of the Expansion bus controller.

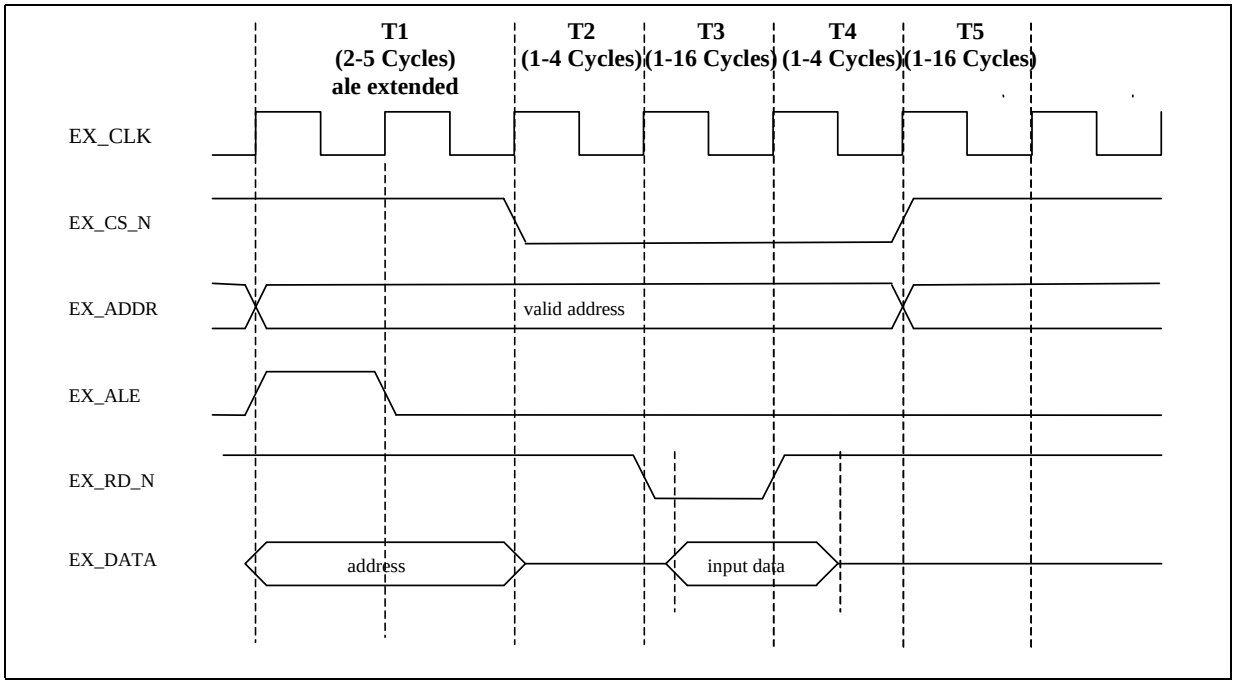
42.4.1.7.1 Intel, Multiplexed-Mode Write Access

Figure 42-6. Expansion-Bus Write (Intel, Multiplexed Mode)



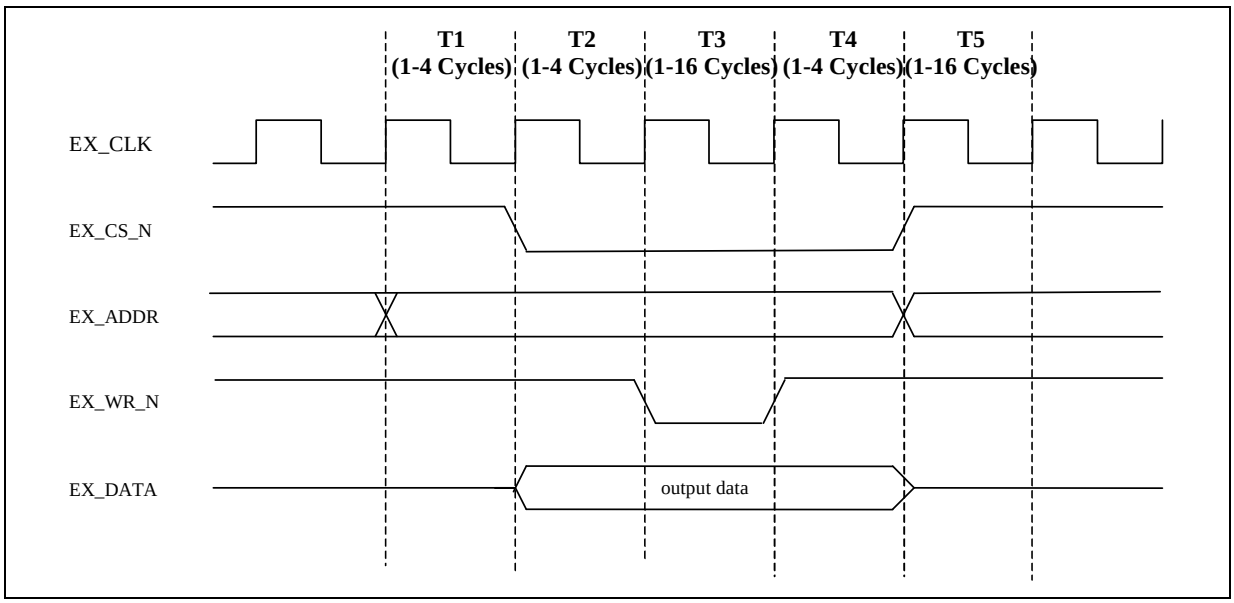
42.4.1.7.2 Intel, Multiplexed-Mode Read Access

Figure 42-7. Expansion-Bus Read (Intel, Multiplexed Mode)



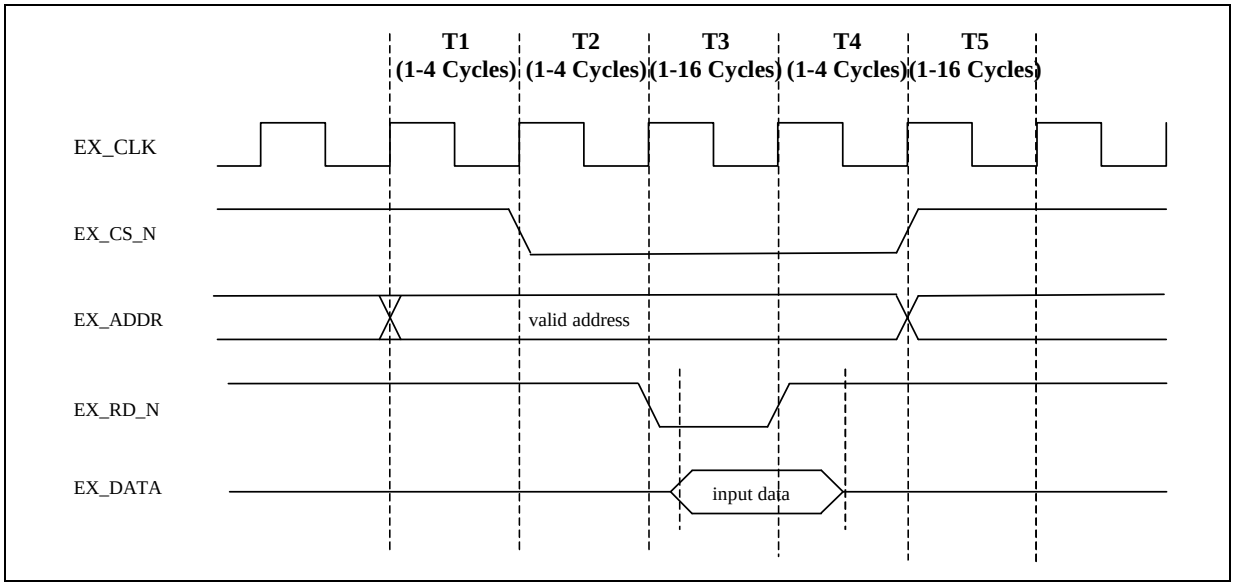
42.4.1.7.3 Intel-Simplex-Mode and Synchronous Intel Write Access

Figure 42-8. Expansion-Bus Write (Intel-Simplex Mode, Synchronous Intel)



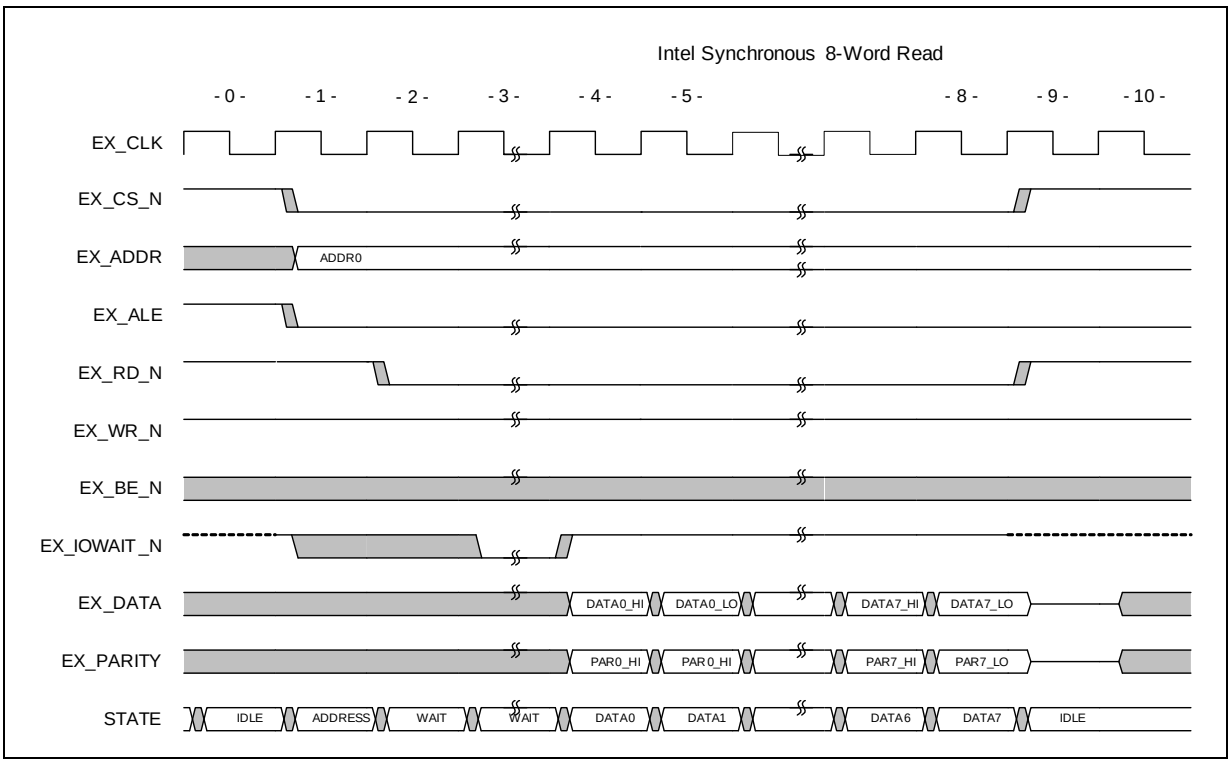
42.4.1.7.4 Intel, Simplex-Mode Read Access

Figure 42-9. Expansion-Bus Read (Intel, Simplex Mode)



42.4.1.7.5 Synchronous Intel 8-word Read Access

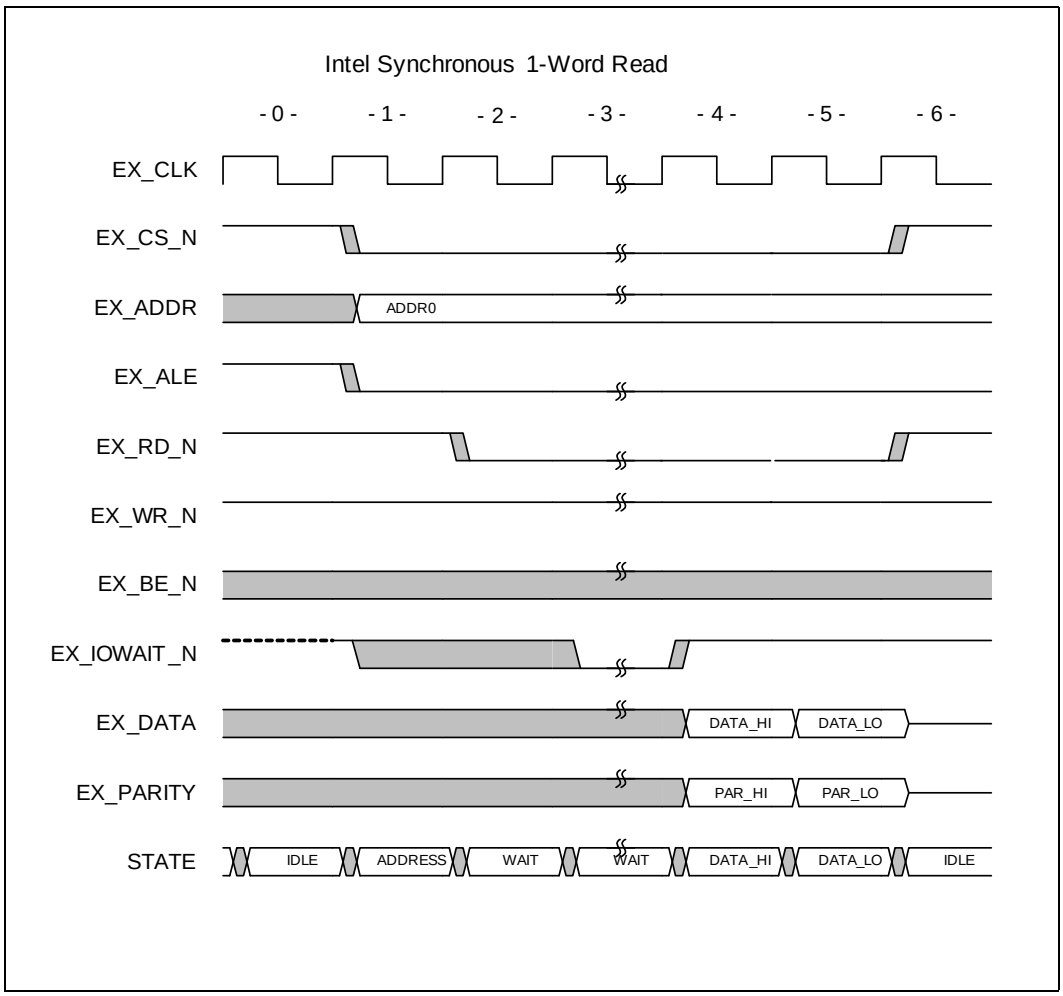
Figure 42-10.Intel Synchronous 8-word Read



The above timing diagram shows an 8-word read to a Synchronous Intel device such as Synchronous Intel StrataFlash. Depending on the EX_CLK period, the latency count bits in the Intel Synchronous Device read configuration register needs to be programmed appropriately based on the timing parameters for the specific device. The Expansion bus controller will always wait in cycles 1 and 2, regardless of EX_IOWAIT_N. The device will then assert EX_IOWAIT_N for several cycles and deassert EX_IOWAIT_N when its ready to transfer data. After the device deasserts EX_IOWAIT_N, it will transfer the remaining words until all 8 words are transferred. The Expansion bus controller and Synchronous Intel device both support wrapping for 8-word reads, therefore ADDR0 is not always aligned to an 8-word boundary. The STATE signal shows the internal Expansion bus state.

42.4.1.7.6 Synchronous Intel 1-word Read Access

Figure 42-11. Intel Synchronous One-Word Read



42.4.1.7.7 Micron* ZBT Write/Read/Write Access

Figure 42-12.Micron* ZBT Write/Read/Write

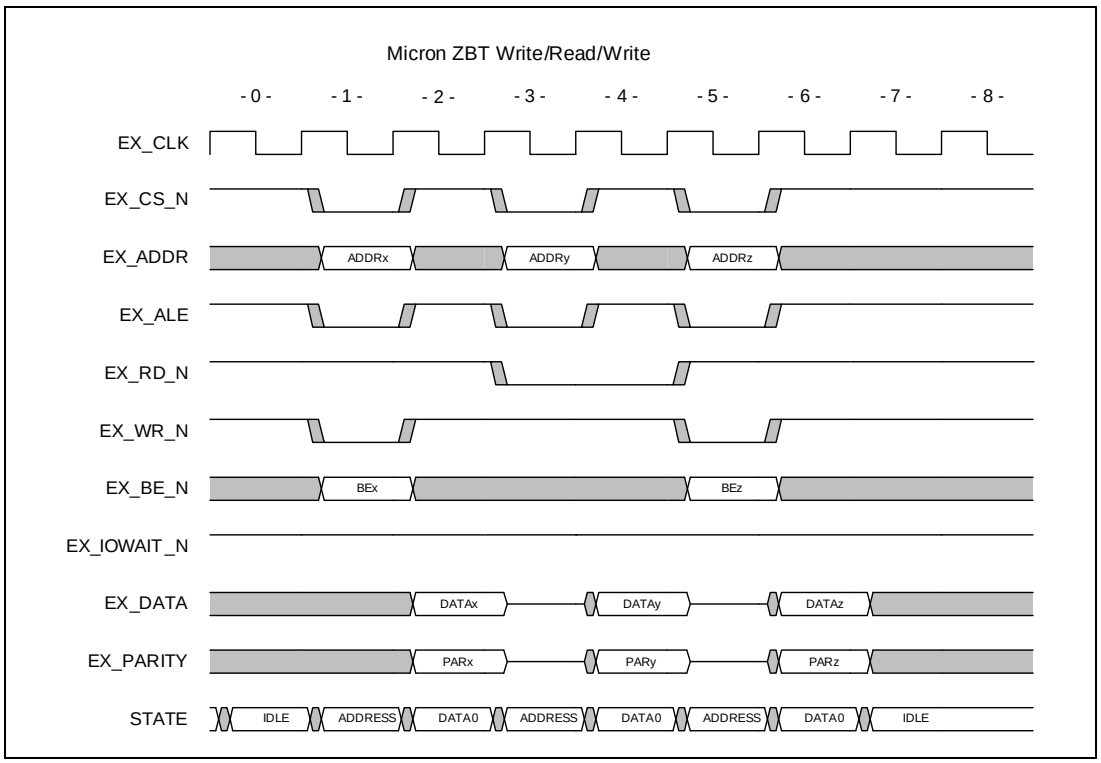
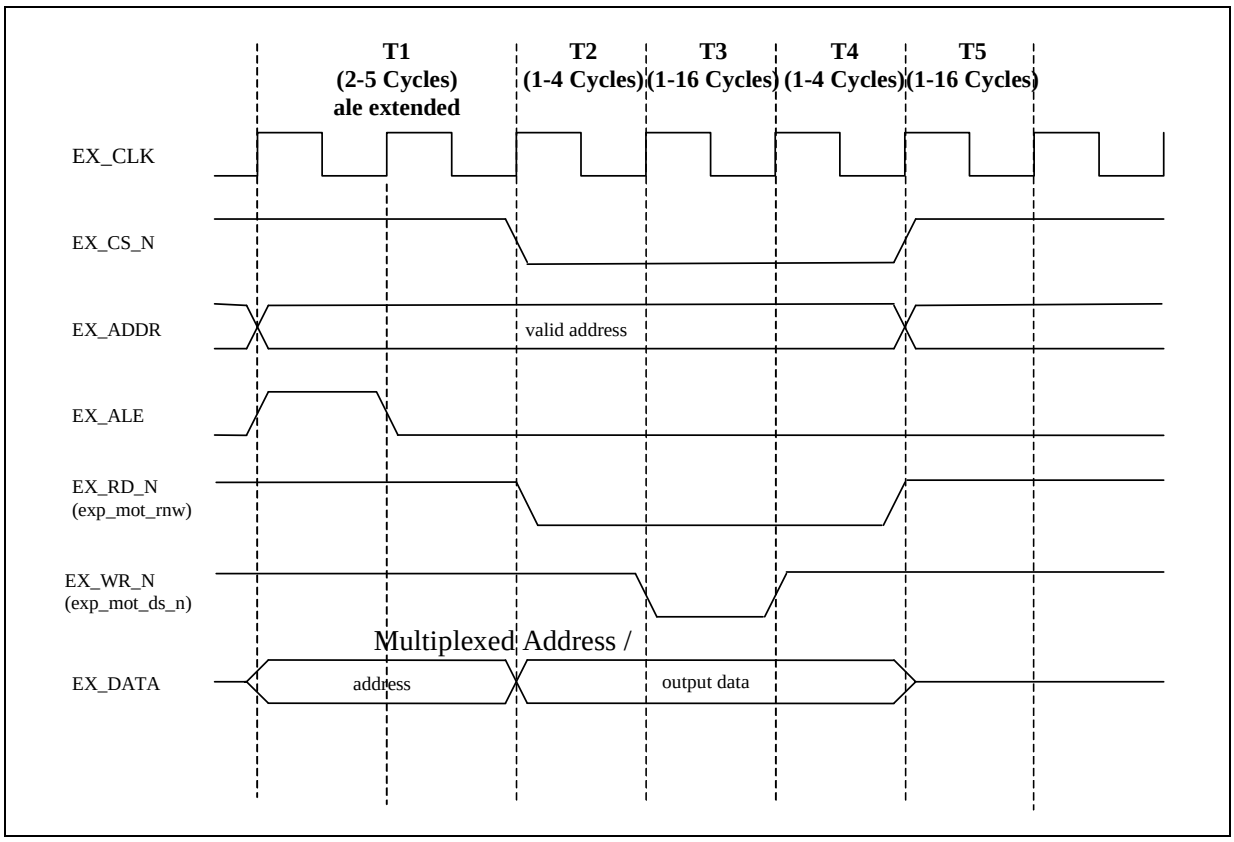


Figure 42-12 shows a write to a Micron ZBT device followed by a read and then followed by another read. The device will never assert EX_IOWAIT_N for this mode.

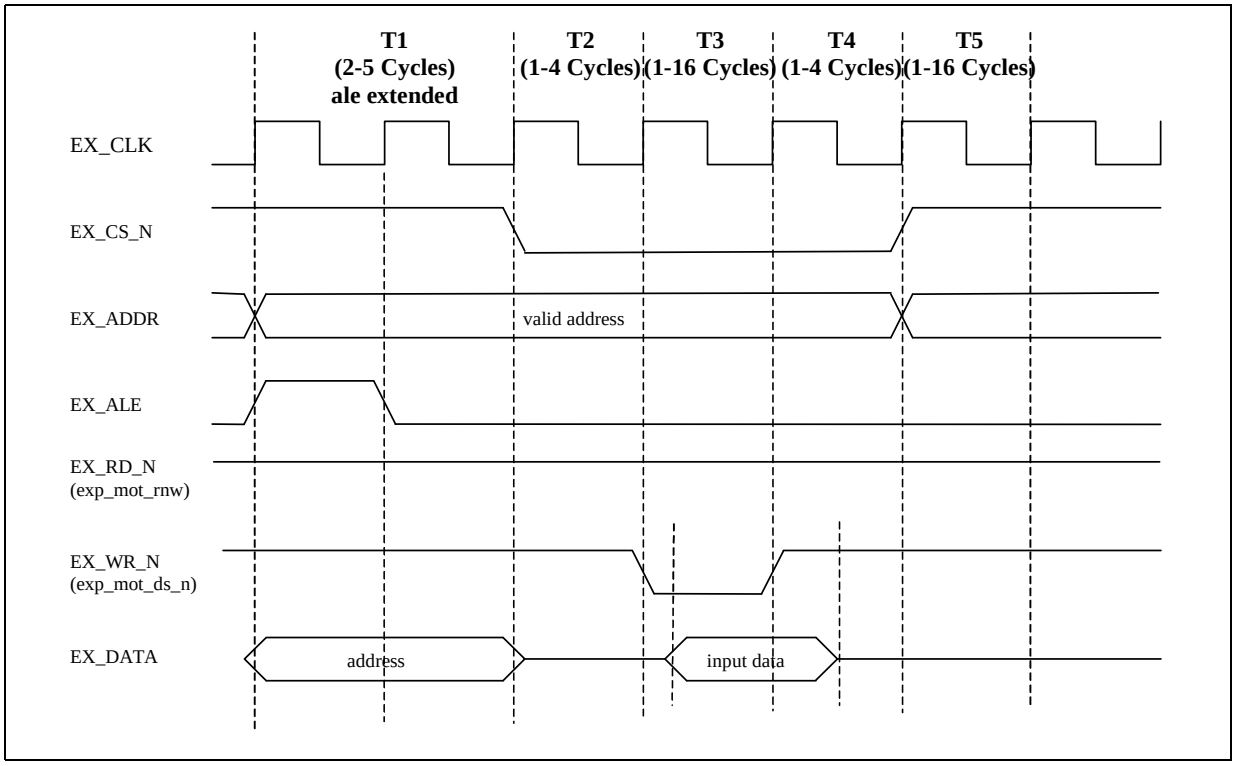
42.4.1.7.8 Motorola*, Multiplexed-Mode Write Access

Figure 42-13.Expansion-Bus Write (Motorola*, Multiplexed Mode)



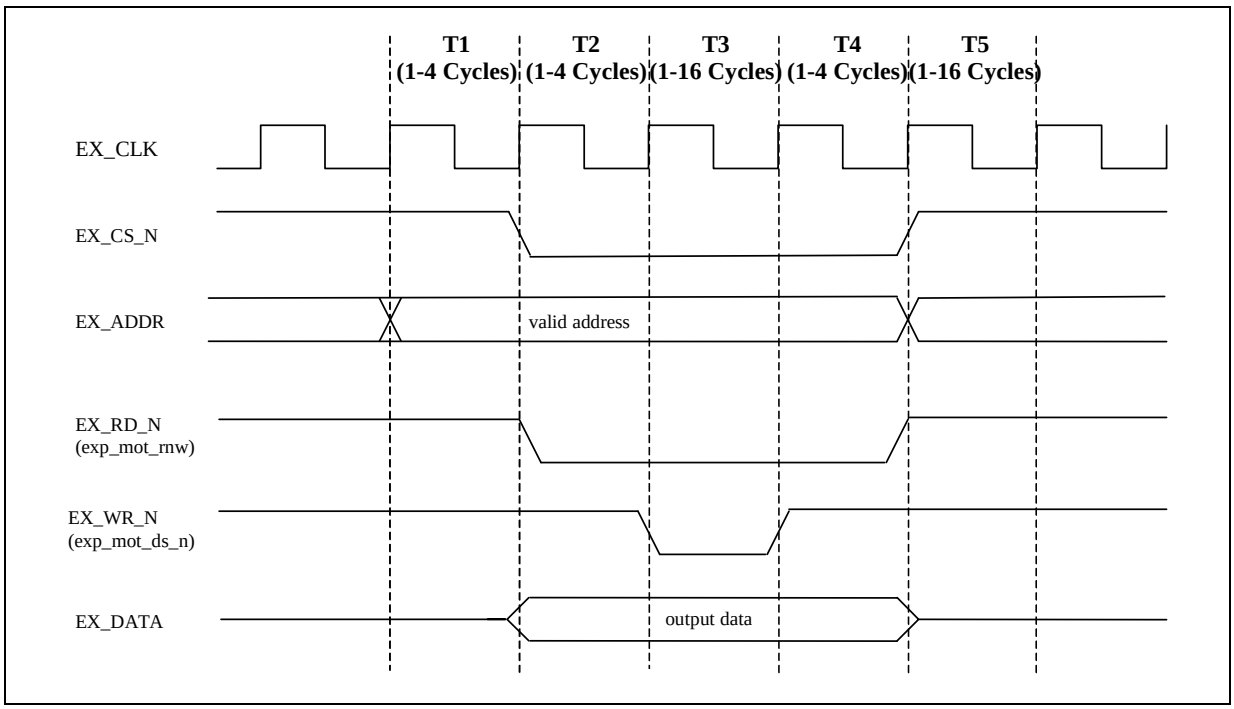
42.4.1.7.9 Motorola*, Multiplexed-Mode Read Access

Figure 42-14.Expansion-Bus Read (Motorola*, Multiplexed Mode)



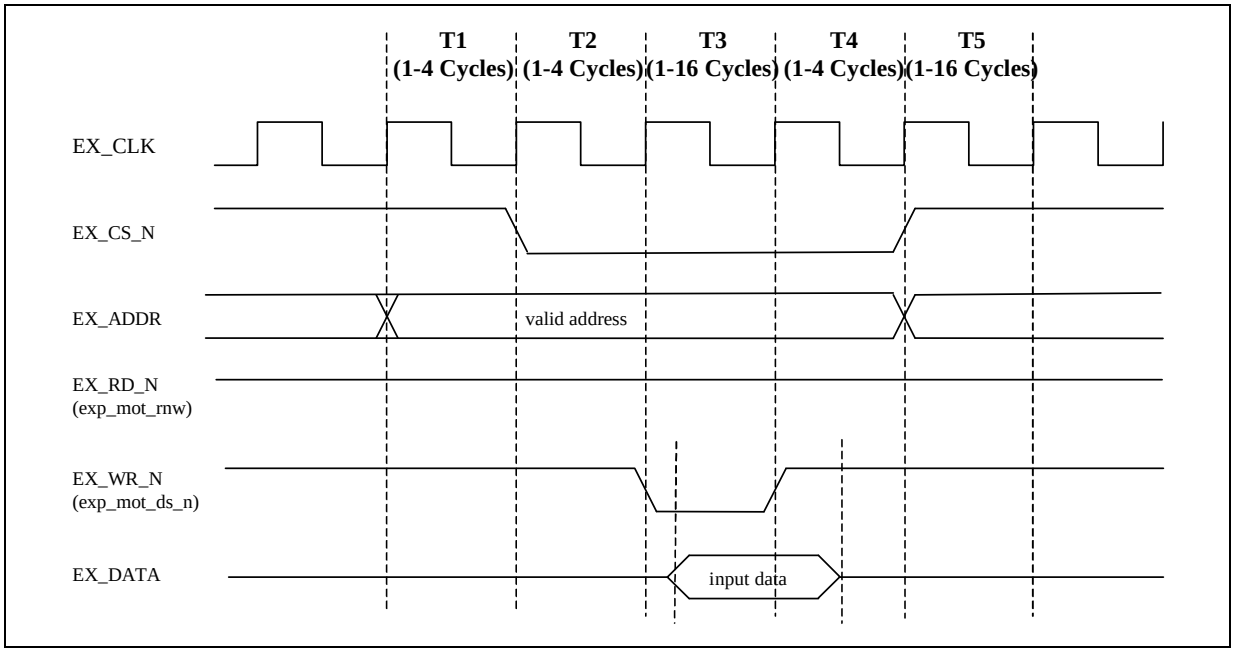
42.4.1.7.10 Motorola*, Simplex-Mode Write Access

Figure 42-15.Expansion-Bus Write (Motorola*, Simplex Mode)



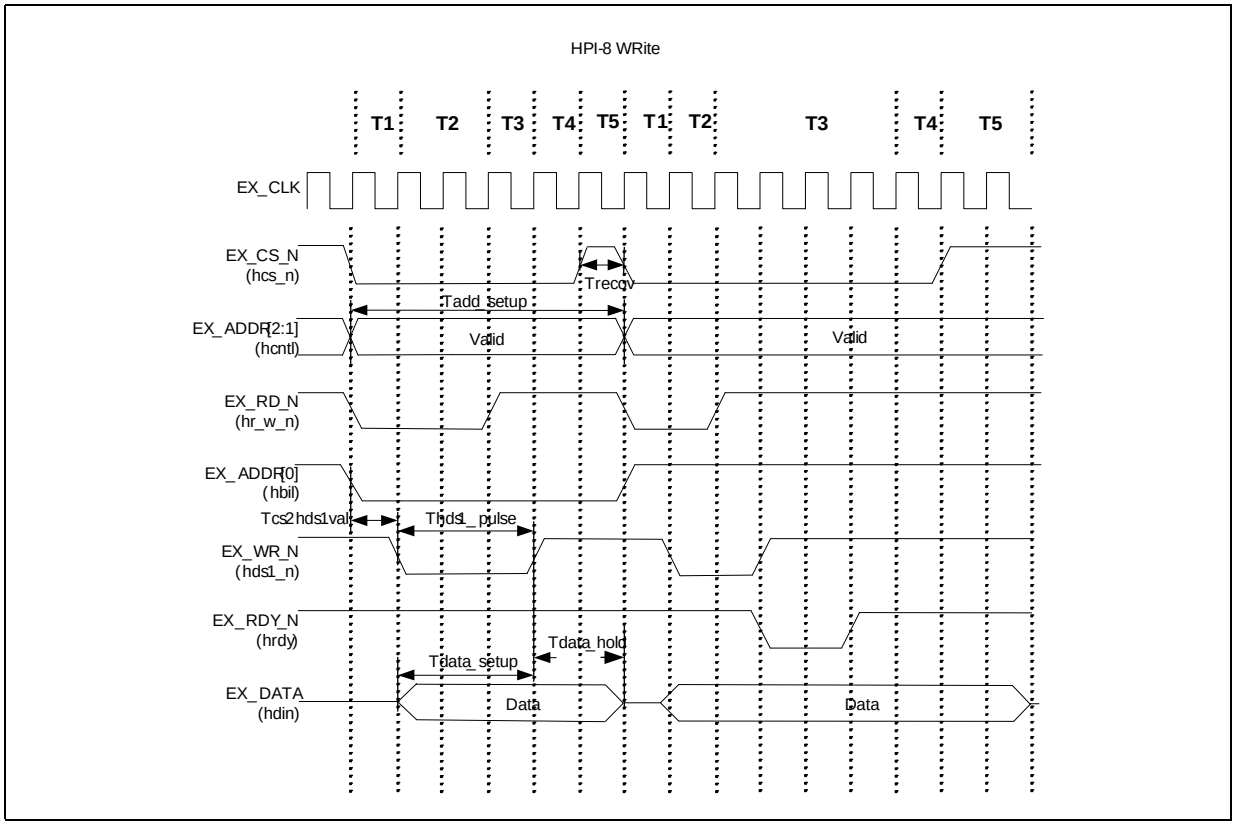
42.4.1.7.11 Motorola*, Simplex-Mode Read Access

Figure 42-16.Expansion-Bus Read (Motorola*, Simplex Mode)



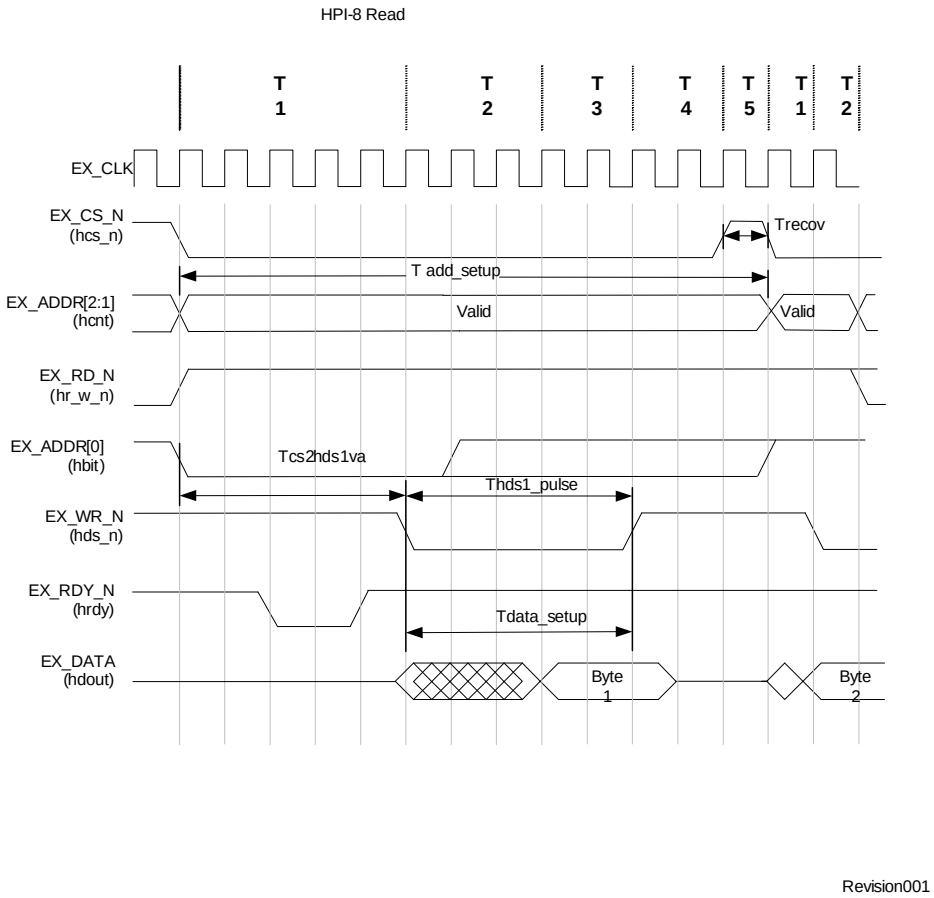
42.4.1.7.12 TI* HPI-8 Write Access

Figure 42-17.Expansion-Bus Write (TI* HPI-8 Mode)



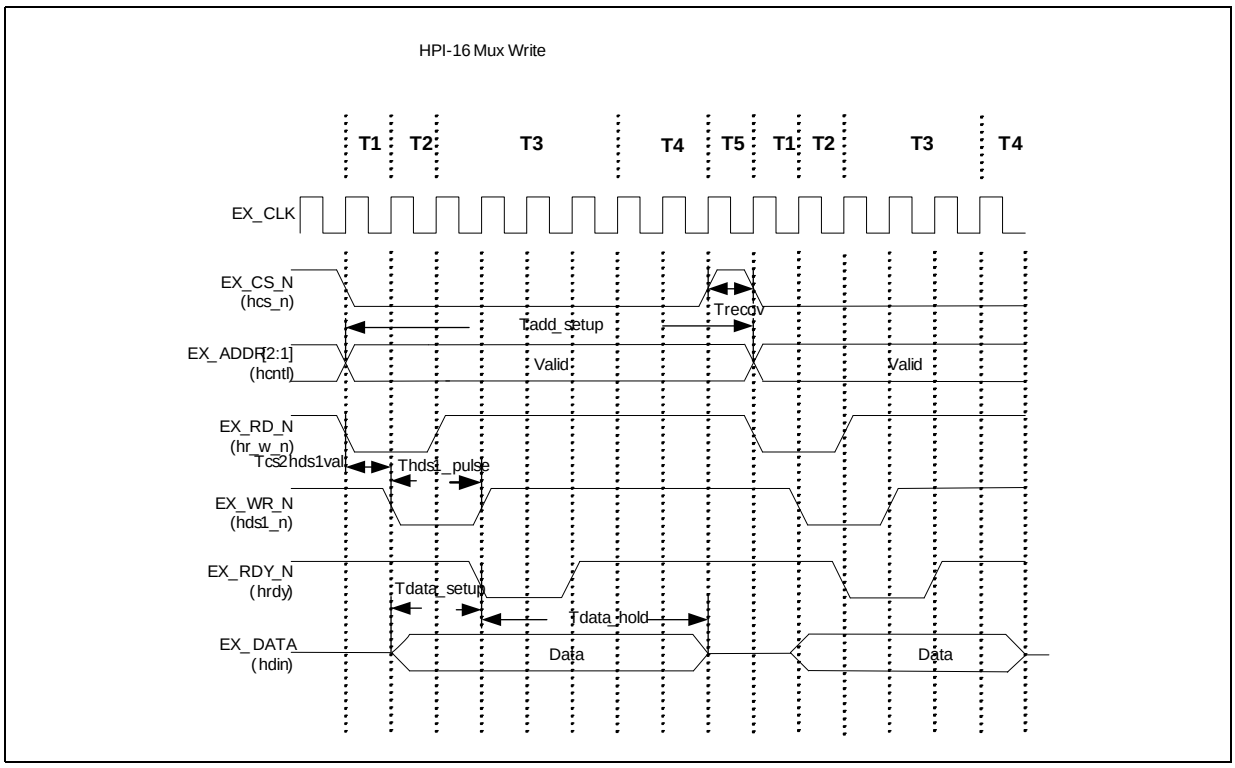
42.4.1.7.13 TI* HPI-8 Read Access

Figure 42-18.Expansion-Bus Read (TI* HPI-8 Mode)



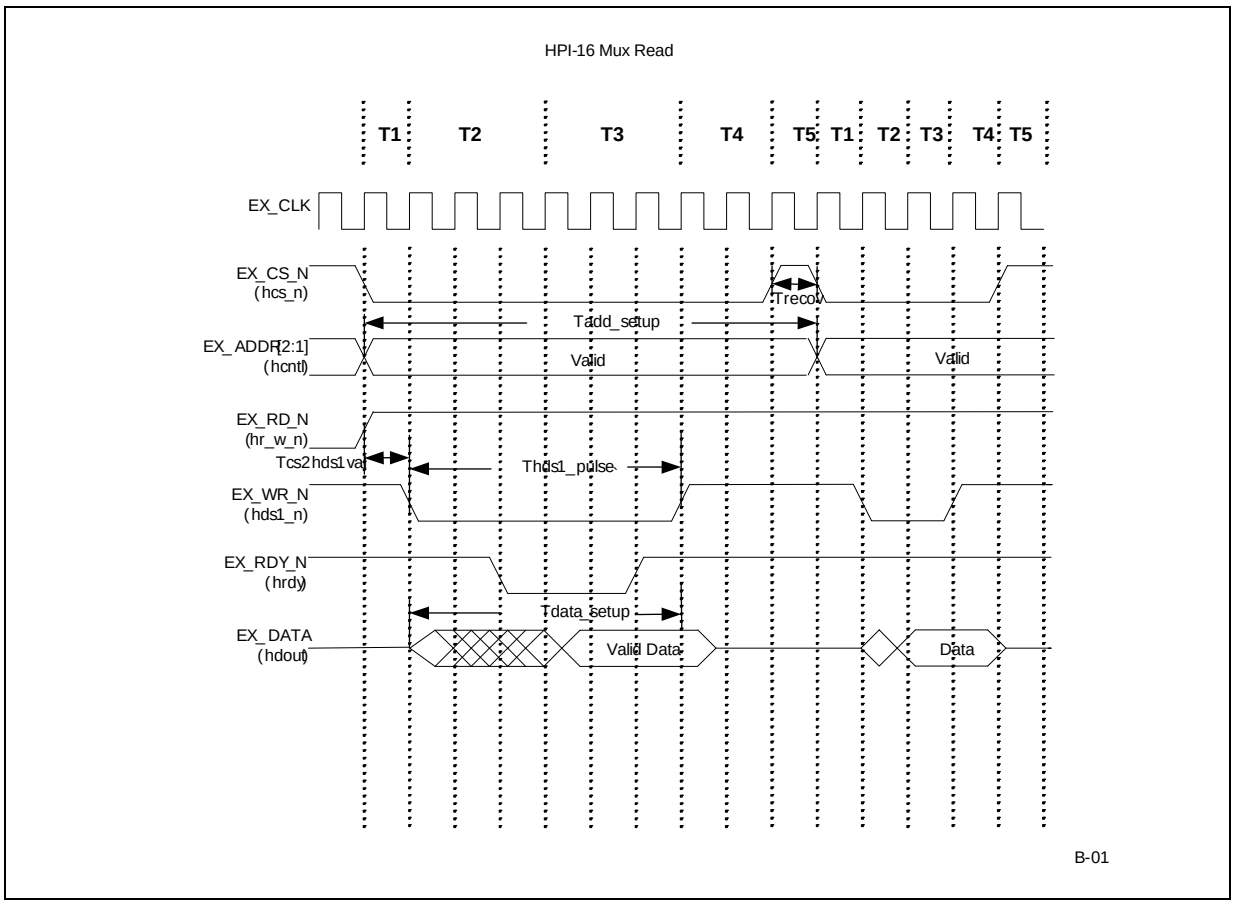
42.4.1.7.14 TI* HPI-16, Multiplexed-Mode Write Access

Figure 42-19.Expansion-Bus Write (TI* HPI-16, Multiplexed Mode)



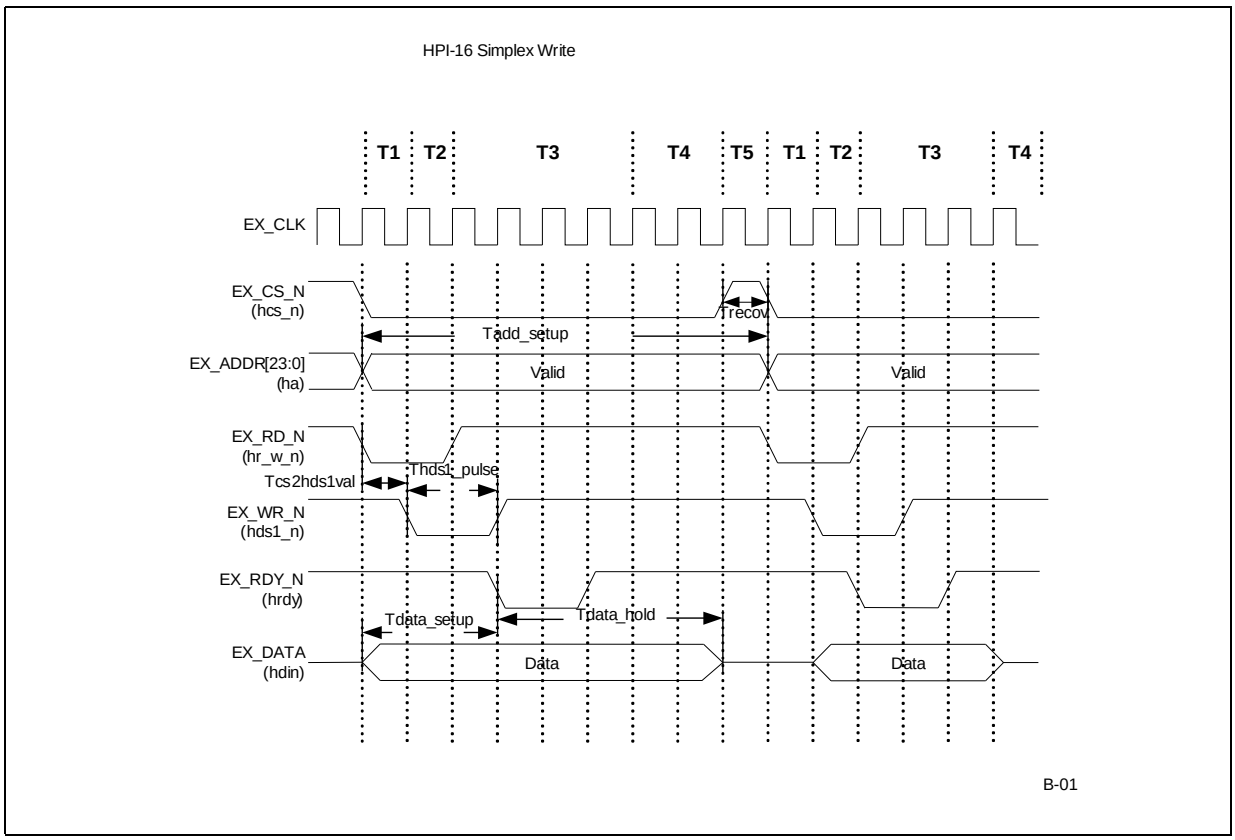
42.4.1.7.15 TI* HPI-16, Multiplexed-Mode Read Access

Figure 42-20.Expansion-Bus Read (TI* HPI-16, Multiplexed Mode)



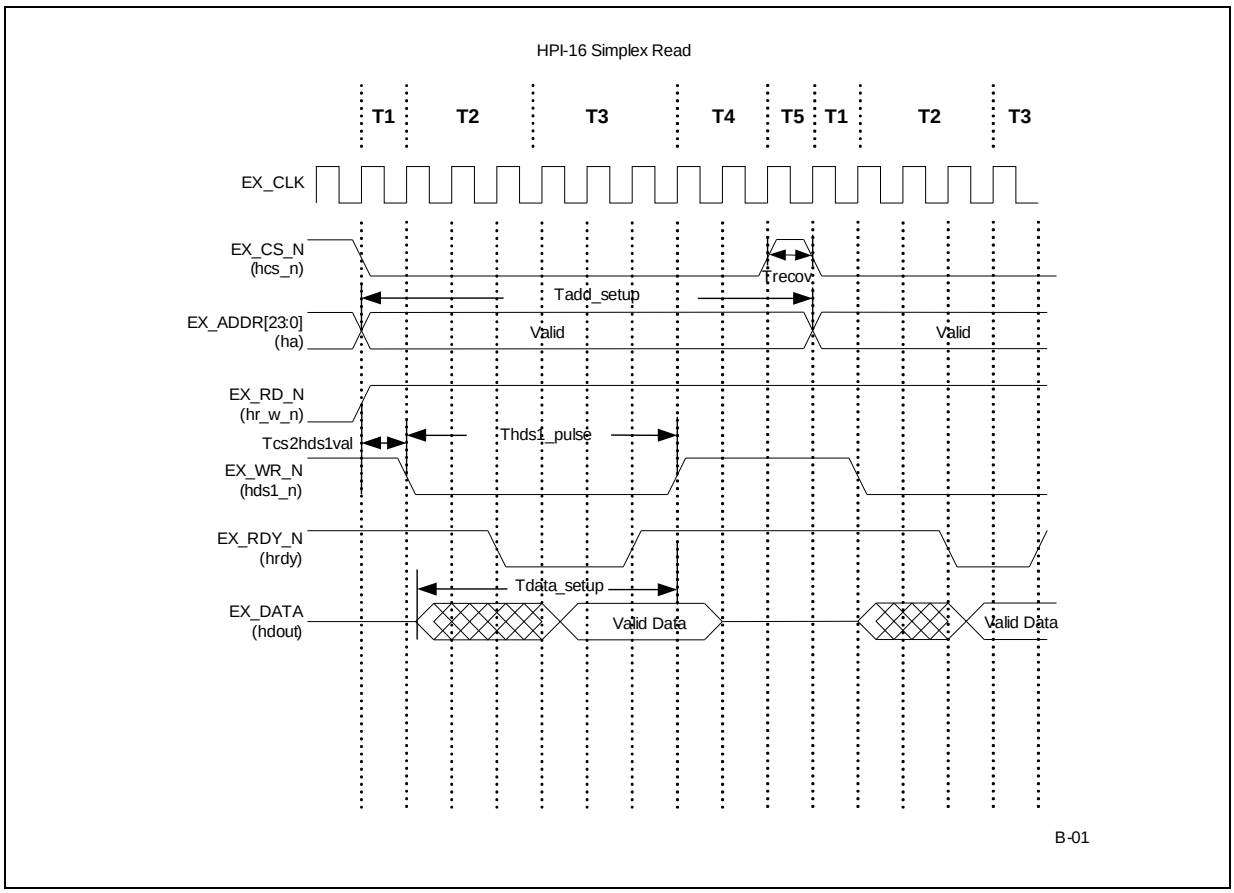
42.4.1.7.16 TI* HPI-16, Simplex-Mode Write Access

Figure 42-21.Expansion-Bus Write (TI* HPI-16, Simplex Mode)



42.4.1.7.17 TI* HPI-16, Simplex-Mode Read Access

Figure 42-22.Expansion-Bus Read (TI* HPI-16, Simplex Mode)



42.5 Register Summary

Accesses to Expansion bus registers are only via 32-bit transfers. Other sized accesses result in unpredictable operation. Accesses to all reserved bits must be written with zero unless otherwise specified. All reserved address spaces must not be read. All writes to reserved address spaces results in unpredictable operation.

For more information on the conventions the following register summaries adopt, see Section 7.1, “Overview of Register Descriptions and Summaries” on page 183.

The Local Expansion Bus registers materialize in the PCI space.

Table 42-5 summarizes the Local Expansion Bus materialization from the PCI perspective.

Table 42-5. Bus M, Device 8, Function 0: Summary of Local Expansion Bus Registers Mapped Through CSRBAR PCI Memory BAR"

Offset Start	Offset End	Register ID - Description	Default Value
00000000h	00000003h	"EXP_TIMING_CS0 - Expansion Bus Timing Register" on page 1698	BFFF3C40h
00000004h at 4h	00000007h at 4h	"EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers" on page 1700	00000000h
00000020h	00000020h	"EXP_CNFG0 -Configuration Register 0" on page 1702	00000040h
00000120h	00000123h	"EXP_PARITY_STATUS - Expansion Bus Parity Status Register" on page 1703	00000000h

42.5.1 Timing and Control Registers

42.5.1.1 EXP_TIMING_CS0 - Expansion Bus Timing Register

The EXP_TIMING_CS registers may only be written if there is not an outstanding Expansion bus transaction. Software must ensure that all outstanding Expansion bus transfers are complete before changing the EXP_TIMING_CS registers.

Table 42-6. EXP_TIMING_CS0 - Expansion Bus Timing Register

Description: Timing and Control Registers					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:8:0	**Offset Start:** 00000000h **Offset End:** 00000003h	
Size: 32 bit	**Default:** BFFF3C40h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	CSx_EN	0 = Chip Select x disabled 1 = Chip Select x enabled		1b	RW
30	PAR_EN	0 = Parity is not generated or compared 1 = Parity is generated and compared Parity is only supported for Intel, Motorola, and Micron ZBT modes.		0b	RW
29 : 28	T1_ADDR_TM	T1 – Address timing 00 = Generate normal address phase timing 01 - 11 = Extend address phase by 1 - 3 clocks		11b	RW
27 :26	T2_SU_CS_TM	T2 – Setup / Chip Select Timing 00 = Generate normal setup phase timing 01 - 11 = Extend setup phase by 1 - 3 clocks		11b	RW
25 : 22	T3_STRB_TM	T3 – Strobe Timing 0000 = Generate normal strobe phase timing 0001-1111 = Extend strobe phase by 1 - 15 clocks		1111b	RW
21 : 20	T4_HOLD_TM	T4 – Hold Timing 00 = Generate normal hold phase timing 01 - 11 = Extend hold phase by 1 - 3 clocks		11b	RW
19 : 16	T5_RCVRY_TM	0000 = Generate normal recovery phase timing 0001-1111 = Extend recovery phase by 1 - 15 clocks		1111b	RW
15 :14	CYC_TYPE	00 = Configures the Expansion bus for Intel cycles. 01 = Configures the Expansion bus for Motorola cycles. 10 = Configures the Expansion bus for HPI cycles. (HPI reserved for chip selects [7:4] only) 11 = Configures the Expansion bus for Micron ZBT cycles		00b	RW
13 : 09	CNFG_4_0	Device Configuration Size. Calculated using the formula: SIZE OF ADDR SPACE = $2^{(9+CNFG[4:1]+16*CNFG[0])}$ For Example: 00000 = Address space of 2^9 = 512 Bytes 00010 = Address space of 2^{10} = 1024 Bytes ... 10000 = Address space of 2^{17} = 128 Kbytes ... 11100 = Address space of 2^{23} = 8 Mbytes 11110 = Address space of 2^{24} = 16 Mbytes XXXX1 = Address space of 2^{25} = 32Mbytes		11110b	RW

Table 42-6. EXP_TIMING_CS0 - Expansion Bus Timing Register

Description: Timing and Control Registers					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:8:0	**Offset Start:** 00000000h **Offset End:** 00000003h	
Size: 32 bit	**Default:** BFFF3C40h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	Sync_Intel	Synchronous Intel StrataFlash® select. This bit must be 0 if CYC_TYPE is not programmed to Intel cycles. 0 = Target device is not a Synchronous Intel StrataFlash 1 = Target device is a Synchronous Intel StrataFlash		0b	RW
07	EXP_CHIP	0 = Target device is not an EP80579 1 = Target device is an EP80579. This bit must only be set to 1 when CYC_TYPE is configured to be Intel Cycles and Sync_Intel is set to 0.		0b	RW
06	BYTE_RD16	Byte read access to Half Word device 0 = Byte access disabled. 1 = Byte access enabled.		1b	RW
05	HRDY_POL	HPI HRDY polarity (reserved for exp_cs_n[7:4] only) 0 = Polarity low true. 1 = Polarity high true.		0b	RW
04	MUX_EN	0 = Separate address and data buses. 1 = Multiplexed address / data on data bus.		0b	RW
03	SPLT_EN	0 = Internal Bus split transfers disabled. 1 = Internal Bus split transfers enabled.		0b	RW
02	Reserved	Reserved. This bit must be written with a '0'. Writing a '1' will result in unpredictable behavior.		0b	RW
01	WR_EN	0 = Writes to CS region are disabled. 1 = Writes to CS region are enabled.		0b	RW
00	BYTE_EN	0 = Expansion bus uses 16-bit-wide data bus 1 = Expansion bus uses only 8-bit data bus		0b	RW

42.5.1.2 EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers

The EXP_TIMING_CS registers may only be written if there is not an outstanding Expansion bus transaction. Software must ensure that all outstanding Expansion bus transfers are complete before changing the EXP_TIMING_CS registers.

Table 42-7. EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers (Sheet 1 of 2)

Description: Timing and Control Registers					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:8:0	**Offset Start:** 00000004h at 4h **Offset End:** 00000007h at 4h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	CSx_EN	0 = Chip Select x disabled 1 = Chip Select x enabled		0h	RW
30	PAR_EN	0 = Parity is not generated or compared 1 = Parity is generated and compared Parity is only supported for Intel, Motorola, and Micron ZBT modes.		0h	RW
29 : 28	T1_ADDR_TM	T1 – Address timing 00 = Generate normal address phase timing 01 - 11 = Extend address phase by 1 - 3 clocks		0h	RW
27 :26	T2_SU_CS_TM	T2 – Setup / Chip Select Timing 00 = Generate normal setup phase timing 01 - 11 = Extend setup phase by 1 - 3 clocks		0h	RW
25 : 22	T3_STRB_TM	T3 – Strobe Timing 0000 = Generate normal strobe phase timing 0001-1111 = Extend strobe phase by 1 - 15 clocks		0h	RW
21 : 20	T4_HOLD_TM	T4 – Hold Timing 00 = Generate normal hold phase timing 01 - 11 = Extend hold phase by 1 - 3 clocks		0h	RW
19 : 16	T5_RCVRY_TM	T5 – Recovery Timing 0000 = Generate normal recovery phase timing 0001-1111 = Extend recovery phase by 1 - 15 clocks		0h	RW
15 :14	CYC_TYPE	00 = Configures the Expansion bus for Intel cycles. 01 = Configures the Expansion bus for Motorola cycles. 10 = Configures the Expansion bus for HPI cycles. (HPI reserved for chip selects [7:4] only) 11 = Configures the Expansion bus for Micron ZBT cycles		0h	RW
13 : 09	CNFG_4_0	Device Configuration Size. Calculated using the formula: SIZE OF ADDR SPACE = $2^{(9+CNFG[4:1]+16*CNFG[0])}$ For Example: 00000 = Address space of 2^9 = 512 Bytes 00010 = Address space of 2^{10} = 1024 Bytes ... 10000 = Address space of 2^{17} = 128 Kbytes ... 11100 = Address space of 2^{23} = 8 Mbytes 11110 = Address space of 2^{24} = 16 Mbytes XXXX1 = Address space of 2^{25} = 32Mbytes		0h	RW

Table 42-7. EXP_TIMING_CS[1-7] - Expansion Bus Timing Registers (Sheet 2 of 2)

Description: Timing and Control Registers					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:8:0	**Offset Start:** 00000004h at 4h **Offset End:** 00000007h at 4h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
08	Sync_Intel	Synchronous Intel StrataFlash® select. This bit must be 0 if CYC_TYPE is not programmed to Intel cycles. 0 = Target device is not a Synchronous Intel StrataFlash 1 = Target device is a Synchronous Intel StrataFlash		0h	RW
07	EXP_CHIP	0 = Target device is not an EP80579 1 = Target device is an EP80579. This bit must only be set to 1 when CYC_TYPE is configured to be Intel Cycles and Sync_Intel is set to 0.		0h	RW
06	BYTE_RD16	Byte read access to Word device 0 = Byte access disabled. 1 = Byte access enabled.		0h	RW
05	HRDY_POL	HPI HRDY polarity (reserved for exp_cs_n[7:4] only) 0 = Polarity low true. 1 = Polarity high true.		0h	RW
04	MUX_EN	0 = Separate address and data buses. 1 = Multiplexed address / data on data bus.		0h	RW
03	SPLT_EN	0 = Internal Bus split transfers disabled. 1 = Internal Bus split transfers enabled.		0h	RW
02	Reserved	Reserved. This bit must be written with a '0'. Writing a '1' will result in unpredictable behavior.		0h	RW
01	WR_EN	0 = Writes to CS region are disabled. 1 = Writes to CS region are enabled.		0h	RW
00	BYTE_EN	0 = Expansion bus uses 16-bit-wide data bus 1 = Expansion bus uses only 8-bit data bus		0h	RW

Note: The seven EXP_TIMING_CS[1-7] registers have following starting offsets:

00000004h - EXP_TIMING_CS1

00000008h - EXP_TIMING_CS2

0000000Ch - EXP_TIMING_CS3

00000010h - EXP_TIMING_CS4

00000014h - EXP_TIMING_CS5

00000018h - EXP_TIMING_CS6

0000001Ch - EXP_TIMING_CS7

42.5.2 Configuration and Status Registers

42.5.2.1 EXP_CNFG0 - Configuration Register 0

At power up or whenever RESET_IN_N is asserted, the Expansion-bus address outputs are switched to inputs and the states of the bits are captured and stored in Configuration Register 0. This occurs when RESET_OUT_N is deasserted.

Only bits EX_ADDR[23:21] are used.

These configuration bits are made available to the system as outputs from the Expansion bus controller block. The bits are read-only.

Table 42-8. EXP_CNFG0 -Configuration Register 0

Description:	Defines the reset-time configuration straps.				
View: PCI	**BAR:** CSRBAR	**Bus:Device:Function:** M:8:0		**Offset Start:** 00000020h **Offset End:** 00000020h	
Size: 32 bit	**Default:** 00000040h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 :24	RSVD	Reserved		0h	RO
23 :21	LEB_SIZE	These bits are initialized from EX_ADDR[23:21] at reset using the reset-time strapping mechanism described above. This field always contains the value read from the straps at reset and is **not** updated in the event BIOS over-rides the value of LEB_SIZE. The effect of these bits is defined in Table 35-159, "MMBAR ADDR Field Behavior" on page 1325		X	RO
20 :11	RSVD	Reserved		0h	RO
10	EXIOW	1 = EX_IOWAIT_N is sampled during the read/write Expansion bus cycles as defined in Section 42.4.1.4, "Using I/O Wait" on page 1680 for Chip Select 0. 0 = EX_IOWAIT_N is ignored for read and write cycles to Chip select 0 if EXP_TIMING_CS0 is configured to Intel mode. Typically, IOWAIT_CS0 must be pulled down to Vss when attaching a Synchronous Intel StrataFlash on Chip select 0 since the default mode for EXP_TIMING_CS0 is Intel mode and EX_IOWAIT_N is an unknown value for Synchronous Intel StrataFlash. If the board does not connect the Synchronous Intel StrataFlash WAIT pin to EX_WAIT_N (and the board guarantees EX_IOWAIT_N is pulled up), the value of IOWAIT_CS0 is a don't care since EX_IOWAIT_N will not be asserted. When EXP_TIMING_CS0 is reconfigured to Intel Synchronous mode during boot-up (for Synchronous Intel chips), the Expansion bus controller ignores EX_IOWAIT_N during read and write cycles since the WAIT functionality is determined from the EXP_SYNCINTEL_COUNT and EXP_TIMING_CS registers.		0h	RO
9 :7	RSVD	Reserved		0h	RO
6	RSVD	Reserved		1h	RO
5	RSVD	Reserved		0h	RO
4	RSVD	Reserved		0h	RW
3 :0	RSVD	Reserved		0h	RO

42.5.2.2 EXP_PARITY_STATUS - Expansion Bus Parity Status Register

Table 42-9. EXP_PARITY_STATUS - Expansion Bus Parity Status Register

Description: Specifies the parity error status.					
View: PCI	**BAR:** CSRBAR		**Bus:Device:Function:** M:8:0	**Offset Start:** 00000120**h** **Offset End:** 00000123h	
Size: 32 bit	**Default:** 00000000h			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31 : 02	ErrorAddr	Address corresponding to the parity error For Outbound transactions, the derivation of this address is as follows: • ErrorAddr = Internal_Bus_ADDR[31:5] & EX_ADDR[4:2] Internal_BusInternal_BusInternal_BusInternal_BusIf multiple parity errors occur (eg- multiple parity errors within a multi-word transfer/burst; or additional parity errors detected on subsequent transactions), then ErrorAddr contains the address of the first parity error. After receiving a parity error, the ErrorAddr is locked until OutErrorSts is cleared by software. If another parity error occurs prior to the interrupt and EXP_PARITY_STATUS being cleared, then the subsequent parity errors will not be trapped nor will the address be logged. As a result, only the first errored transaction will be dropped, but the subsequent (errored) transaction(s) will be completed. Note: ErrorAddr will not be unlocked until the current transaction is complete.		0h	RO
01	RSVD	Reserved		0h	RWC
00	OutErrorSts	A parity error occurred on an outbound read. This bit will not get set unless PAR_EN is set in the EXP_TIMING_CS for which the parity error has occurred.		0h	RWC

OutErrorSts is generated to forms exp_parity_error which is routed to the Interrupt Controller and can generate interrupts to the processor core. In the event of multiple outbound parity errors, there is a race condition between when software performs a write to clear the EXP_PARITY_STATUS register and setting the InErrorSts/OutErrorSts error bit in hardware.

If the software clears the error bit before another parity error, the EXP_PARITY_STATUS register will be set again. However if software clears the error bit on or after another parity error, the EXP_PARITY_STATUS register will be cleared.

42.6 Performance Estimation

The bandwidth that can be supported on the expansion bus depends on the latencies through the system and the nature of the protocol on the expansion bus. All latencies are dependent on whether the transaction is read or write.

Each specific protocol supported by the expansion has it's own set of timing parameters as described by the number cycles required for T1, T2, T3, T4 and T5. The reader is referred to Section 42.4, "Theory of Operation" for explanation of these parameters. Additionally, the number of wait states required contributes additional cycles required for a given transaction and will reduce the bandwidth.

Estimated system latencies expressed in terms of Expansion Bus clock cycles are shown in Table 42-10.

The estimated bandwidth for a given platform configuration is determined as:

BW (Mbit/s) = TransactionSize * LEB_Freq / TotalLatency

where:

- TransactionSize is size of the transaction expressed in bits
- LEB_Freq is the frequency of the external clock being used to clock the Expansion bus
- TotalLatency is the sum of:
  - SytemLatency + T1 + T2 + T3 + T4 + T5 + Wait States

Note: The appropriate values for these latencies must be used dependending on whether the transaction is a read or a write.

Note: This estimate applies to a single, serialized request at the entry to the AIOC (PCI-to-PCI Bus M Bridge) onto the expansion bus

Table 42-10. LEB Performance Calculation - Estimated AIOC Latencies

AIOC Internal Bus (MHz)	LEB (MHz)	AIOC Latency Read (LEB cycles)	AIOC Latency Write (LEB cycles)
400	80	31	27
533	80	24	21
800	80	22	19
400	66	26	22
533	66	19	17
800	66	17	15

Note: Note: The system is assumed to be otherwise quiescent. No competing traffic on the internal buses is considered.

Note: These latencies comprehend only those from the time an IA initiated transaction is received at the AIOC to the time the AIOC completes the transaction. No IA, FSB, MCH, software, etc latencies are included.

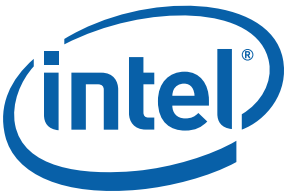
Table 42-11 provides example estimates for various platform configurations.

Table 42-11. Outbound Performance Estimation Examples

	Example A Write	Example A Read		Example B Write	Example B Read		Example C Write	Example C Read
Expansion Bus (MHz)	80	80		80	80		80	80
AIOC Internal Bus (MHz)	400	400		533	533		800	800
Transaction Size(bits)	32	32		32	32		32	32
AIOC Latency	27	31		21	24		19	22
Wait states	0	0		0	0		0	0
T1	1	1		1	1		1	1
T2	2	3		2	3		2	3
T3	1	2		1	2		1	2
T4	1	1		1	1		1	1
T5	1	1		1	1		1	1
Total Latency	33	39		27	32		25	30
Mbits/s	77	66		95	80		102	85

Note: These examples assume 0 wait states introduced by the external LEB device. The latency of the attached device must be taken into account for specific applications.

§ §



Test and Debug, Volume 5 of 6

Order Number: 320066-003US

43.0 Global Design for Test Features

43.1 JTAG

43.1.1 JTAG Functions Overview

The JTAG functionality, has the following attributes.

- The JTAG TAP controller is compliant with the *IEEE 1149.1 Specification*.
- The JTAG block supports the TAP controllers in IA-32 core and IMCH.
- The JTAG block supports the IO Boundary Scan Requirement implementing all the mandatory and some optional instructions from the IEEE 1149.1 specification.

43.1.2 EP80579 TAP Controllers

The JTAG Test Access Port (TAP) controller requires 4 **input** pins: Test Reset (**TRST#**, active Low), Test Clock (**TCK**), Test Mode Select (**TMS**) and the serial Test Data In (**TDI**). TMS controls tSoChe TAP state transitions. The TAP has one **output** pin, the serial Test Data Out (**TDO**).

43.1.2.1 IA-32 Core

There are two JTAG Test Access Port (TAP) controllers: one in the IA-32 core, and one “master” TAP in the MCH. The IA-32 Core TAP is present as a result of the SoC integration and performs no JTAG functions besides BYPASS and IDCODE.

43.1.2.2 MCH TAP Extension

The TAP controller in the IMCH (in the CMI), is the master. It executes all of the IEEE 1149/1 compliant JTAG and Boundary Scan functions in the EP80579.

Serial TAPs have the issue that they have a BYPASS register length >1. This is not desirable from a IEEE 1149.1 compliance standpoint. To get around this, two BSDLs are published, one each for the IMCH and IA-32 core. The IMCH TAP acts as the master TAP, supporting the public instructions for customer usage. The IA-32 core TAP’s public Boundary Scan instructions, while still supported and available, do not perform any meaningful function since the IA-32 core's TAP does not control any IO pins directly. The appropriate Boundary Scan functionality is performed by the corresponding instructions in the IMCH TAP controller. Both TAPs support the BYPASS and IDCODE instructions. This is summarized in Table 43-1.

Table 43-1. EP80579 TAPs Public Instructions

Public Instruction	MCH TAP Controller	VmC TAP Controller
BYPASS	Supported	Supported
IDCODE	EP80579 IDCODE; unique per EP80579 silicon stepping	IA-32 core IDCODE; fixed for all products using the core
EXTEST	Boundary Scan for EP80579 IOs	Implemented, but not connected to any IO pads

43.1.3 EP80579 JTAG ID Codes

On account of the multiple TAPs, the EP80579 supports and displays multiple IDCODEs. This is described below in Table 43-2.

Table 43-2. EP80579 TAP IDCode Values

JTAG TAP Controller	IDCode	Comments
IMCH (EP80579)	32'h**0E66_0013**	The 4 MSBs are incremented for each EP80579 silicon stepping (major or minor)
IA-32 Core	32'h**0B36_0013**	Fixed for all products and steppings using this core

43.1.4 Special Requirements and Limitations

The JTAG implementation has the following requirements for proper JTAG operation.

- For proper boundary scan operation, the following GbE reset pins need to be in the specified states: **SYS_PWR_OK = 1** and **AUX_PWR_GOOD = 1**. This requirement is true even if the GbE units are disabled. These additional non-JTAG pins must be automatically driven to the proper values by the platform's power-up devices when the EP80579 is powered up for boundary scan testing.
- Several high-speed interfaces and critical system reset and power related pins have been removed from the Boundary Scan chain. These are documented in Section 43.2.1, "JTAG Boundary Scan".

43.1.5 JTAG Instructions Summary: MCH

The table in this section summarizes the EP80579-specific JTAG instructions implemented in the MCH section of the chip.

The JTAG instructions are classified as follows:

- **Public**: These instructions are available to anyone.

Table 43-3. JTAG Instructions Summary for MCH

Opcode (8'Hxx)	Instruction	Register Size	Type	Special Implтn.	Section	Description
00	EXTEST	884	Public	-	Section 45.2.1.1.1	Boundary Scan -- Extest
01	SAMPPRE	884	Public	-	Section 45.2.1.1.2	Boundary Scan -- Sample Preload
02	IDCODE	32	Public	-	Section 45.2.1.1.3	IDCode
03			reserved			not implemented
05-07			Reserved			Not implemented
08	HIGHZ	0	Public	-	Section 45.2.1.1.4	Boundary Scan -- HighZ
09			Reserved			Not implemented
FF	BYPASS	0	Public	-	Section 45.2.1.1.5	Bypass
16-1B			reserved			Not implemented
43-46			Reserved			Not implemented
6C-6F			Reserved			Not implemented

43.2 I/O Testing

The EP80579 implements JTAG Boundary Scan to allow testing of the I/O pins.

43.2.1 JTAG Boundary Scan

The EP80579 implements *IEEE 1149.1 compliant JTAG Boundary Scan (BScan)* on all its interfaces withthe exception noted below.

The three high-speed interfaces, PCIe, SATA and USB2 did not implement JTAG BScan. These are tested with an XOR chain that covers the AFEs of all 3 interfaces (Section 46.2.3, "XOR Chains"). In addition, several compliance pins were excluded from the BScan chain to ensure proper functionality. These are listed in Table 43-4. The Boundary Scan Chain is defined in the EP80579 BSDL file.

43.2.1.1 Pins Excluded from Boundary Scan Chain

Several critical compliance pins were excluded from the Boundary Scan chain. These are listed below in Table 43-4.

Table 43-4. Compliance Pins Excluded from Boundary Scan Chain

Interface	Excluded Pin(s)
RTC	RTCX1, RTCX2, RTEST_N
ICH Misc	INTVRMEN
Power Mgmt	PWROK, RSMRST_N
SMBus	INTRUDER_N
CRU	CLKS100, CLKP100, CLKN100
LEB	EXRCMP, EXRCMN
DDR	D_RCOMPX, D_DDRCRES[2:0], D_DRVCRES, D_SLEWCRES
VC signal	XXTHRMDC_OUT, XXTHRMDA_IN, XXTPCD_OUT, XXTPCL_OUT, XXHFPLL_OUT
GBE	GBE_RCOMPP, GBE_RCOMPN, SYS_PWR_OK, GBE_AUX_PWR_GOOD

§ §

44.0 IA-32 Core

The testability features supported by the IA-32 core integrated into Intel® EP80579 Integrated Processor are described in this section.

44.1 JTAG

The IA-32 core TAP (Test Access Port) complies with the IEEE 1149.1 (“JTAG”) test architecture standard. The TAP implements an instruction set of 6 functions as follows:

- SAMPLE/PRELOAD
- EXTEST
- CLAMP
- HIGHZ
- IDCODE
- BYPASS

44.1.1 Usage

There are seven 1149.1-defined public instructions implemented in the IA-32 core TAP. These instructions are described in Table 44-1. These instructions select from among three different TAP data registers — the boundary scan, device ID, and bypass registers.

Table 44-1. 1149.1 Public Instructions in the IA-32 Core TAP

TAP Instruction (op-code)	Function during StopClk	Data Register Selected	Action during...			
			RT/IDLE	Capture-DR	Shift-DR	Update-DR
SAMPLE/ PRELOAD (0000001)	Yes	Boundary scan	—	Sample all IA-32 core pins	Shift bscan reg	Update bscan reg. outputs
EXTEST (0000000)	Yes	Boundary scan	—	Sample all IA-32 core pins	Shift bscan reg	Update IA-32 core outputs
CLAMP (0000100)	Yes	Bypass	—	Reset bypass reg	Shift bypass reg	—
HIGHZ (0001000)	Yes	Bypass	—	Reset bypass reg	Shift bypass reg	—
IDCODE (0000010)	Yes	Device ID	—	Load IA-32 core ID code	Shift ID reg	—
BYPASS (1111111)	Yes	Bypass	—	Reset bypass reg	Shift bypass reg	—

The operation of these instructions is summarized as follows:

- **SAMPLE/PRELOAD**: This is the Boundary scan command. The IA-32 core operates normally. IA-32 core pins are sampled into the boundary scan register on CaptureDr, shifted out on ShiftDr.
- **EXTEST**: Same as SAMPLE, except the IA-32 core outputs are driven directly from the boundary scan register (so that the external pins wiggle as you shift the results out).
- **CLAMP**: The IA-32 core outputs are driven directly from the previously-loaded value in the boundary scan register; the 1-bit TAP bypass register is connected between TDI and TDO.
- **HIGHZ**: All outputs of the IA-32 core are driven to their high-impedance state.
- **IDCODE**: The IA-32 core operates normally; the Device ID register is read using this TAP instruction. The IDCODE returned contains the fields described in Table 44-2.
- **BYPASS**: The IA-32 core operates normally; TAP pins TDI and TDO access the 1-bit bypass register.

Table 44-2. Device ID Register Bit-fields

Field Name	Bit Position(s)	Value	Function
Version Number	31-28	**xxxx** (0000 for A-0)	What version of IA-32 core this unit is. Should increment with each stepping
Part Number	27-12	**1011001101100000** (Request Number 101100) (Product 11011) (End User/Market 00000)	Number chosen by Intel to describe what part this is.
Manufacturer Identity	11-1	**00000001001**	Identity code that uniquely identifies Intel.
Device ID register implemented	0	**1**	Must always be a 1 if the Device ID register exists.

44.1.1.1 Description

Full details of the operation of these public instructions can be found in the 1149.1 standard. Please note that the op-codes for the public instructions are an Intel standard and do not vary from processor to processor.

The contents of the Device ID register are shown in Table 44-2. Note that the Version Number changes for each stepping. The rest of the fields do not change during the life of the product. IA-32 core IDCODE value is xB360013h, where x represent the stepping number (x=0000 for A-0).

45.0 IMCH Design for Test

45.1 IMCH Design for Test Features

45.1.1 Features

The IMCH includes a number of Design For Test features. These features are described this section.

45.2 JTAG

45.2.1 IMCH JTAG Instructions

Table 45-1 below lists the JTAG chains, their public or private designation, the chain length, and a short descriptive name. The chains are described in more detail in later sections.

Due to the nature of the JTAG design in EP80579, there are two operating modes in which the instructions below are accessible. In the default serial mode (i.e. MCH JTAG + IA-32 core JTAG), the TDI value represents the full input instruction required.

Table 45-1. IMCH JTAG Instructions

Mnemonic	Parallel Mode IR	Hex	Serial Mode IR	Hex	Public	Chain Length	Description
ExTest	00000000	00	000000001111111	007F	Public	884	Extest
SampPre	00000001	01	000000011111111	00FF	Public	884	Sample Preload
IDCode	00000010	02	000000101111111	017F	Public	32	IDCode
HighZ	00001000	08	000010001111111	047F	Public	0	HighZ
ByPass	11111111	FF	111111111111111	7FFF	Public	0	Bypass

45.2.1.1 JTAG Chain Details

Descriptions follow for the JTAG chain.

45.2.1.1.1 ExTest

The ExTest instruction allows circuitry or wiring external to the devices to be tested. Output boundary scan register cells are used to apply stimulus, while boundary scan cells at the inputs pins are used to capture data. I/O cells can be used for either purpose, depending on the value set in the corresponding output enable cell.

Note that some signals are prevented from driving out during ExTest: xxpwrgd, xxrstinn, and xxextintrnn.

45.2.1.1.2 Sample/Preload

The Sample/Preload instruction is used to allow scanning of the boundary scan register without causing interference to the normal operation of the device. Two functions can be performed by use of the Sample/Preload instruction.

Sample - allows a snapshot of the data flowing into and out of the device to be taken without affecting the normal operation of the device.

Preload - allows an initial pattern to be placed into the boundary scan register cells. This allows initial known data to be present prior to the selection of another boundary scan test operation.

45.2.1.1.3 IDCode

This instruction is used to determine the manufacturer, part number, and revision level of the device being accessed. This instruction selects only the device identification register to be connected for serial access between TDI and TDO in the Shift-DR controller state. When this instruction is selected, the operation of the test logic shall have no effect on the operation of the on-chip system logic. Reading the IDCode returns a single value.

The ID consists of 6 fields. The first field is for the version and stepping of the device. The Part Number field contains 3 fields. Field #1 is the product category, and for a server, has a value of “000 100” binary. Field #2 is product type and is “01 000” for the IMCH. Field #3 is an assigned chronological value within a product type. The Manufacturer ID is “000 0000 1001” for Intel as assigned by IEEE. The last field is filled by a binary “1”.

Table 45-2. JTAG Device Identification Register Field Designations

Version/ Stepping	Part Number			Manufacturer ID	Hardwired ‘1’
	Field #1	Field #2	Field #3		
4 bits	6 bits	5 bits	5 bits	11 bits	1 bit

The JTAG 32-bit ID contains information in order to uniquely identify a component, and its manufacturer. Note that this ID does not uniquely distinguish between devices of a given component type. This ID does not contain a device serial number. The fields as defined are shown in the table below.

The revision codes must be changed to match chip specification.

45.2.1.1.4 HighZ

The HIGHZ instruction is used to force all outputs of the device (except TDO) into a high impedance state. This instruction shall select the Bypass Register to be connected between TDI and TDO in the Shift-DR controller state.

45.2.1.1.5 Bypass

The BYPASS command selects the Bypass Register to be connected for serial access between TDI and TDO pins of the device. The Bypass Register is a single bit register and is used to provide a minimum-length serial path through the device. This allows more rapid movement of test data to and from other components in the system.

45.3 High Speed I/O Testing

There is a single XOR chain which connects PCIe pads, along with USB and SATA pads. The definition for this is located in the ICH XOR Chain section (Section 46.2.3, “XOR Chains”).

§ §

August 2009
Order Number: 320066-003US

46.0 ICH Design for Test

46.1 JTAG

The IICH in the EP80579 does not have a JTAG interface.

46.2 I/O Test Mode

46.2.1 Test Mode Entry Methods

Non-functional test modes are dedicated test modes where the chip is not operating in its normal manner. This test mode includes Boundary scan. Functional test modes are features built into the existing functional modes that allow the IICH to be optimally tested and/or margined in its normal operating mode.

46.2.1.1 Non-Functional Test Mode Entry

This section covers the Non-functional Test Mode entry methods.

46.2.1.1.1 Serial Test Mode (STM) Entry

The Non-functional test mode is activated with the Serial Test Mode Entry. This method uses a single test pin (DFXTEST#) to write to a test-mode command register or to read from a test-mode status register.

DFXTEST_N pin on EP80579 is shared between IICH and IMCH. By default it works as DFXTEST_N pin for IICH and if JTAG_INSTR_0x73 is set in the MCH, then it acts as TESTNN pin for IMCH.

The protocol for this serial test mode entry is based on PCICLK and is depicted in Table 46-1. For either a read or a write, an 18-bit serial command field is fed into the test pin, prefaced with a ‘0’ as the start bit (the idle condition of the test pin must be a logic-high). For a write, the 18-bit command field is followed by 32 bits of write data, which is fed into the appropriate test-mode command register. For a read, the 18-bit command field is followed by one clock of turn-around, followed by the test-pin driving out 32 bits of read data. Figure 46-1 and Figure 46-2 demonstrate the timing of the Serial Test Mode Entry read and write cycles. It is important that test modes are not activated until after the entire data field has been written. It is also important that if a register is read, that its value is sampled at the end of the turn-around cycle. All serial values are put out MSB-first.

All STM accessible registers are implemented as STICKY registers.

Table 46-1. Serial Test Mode Entry Command Field

Bit Field	Default and Access	Description
[17:13]	00000b – Write-Only	Register Index (31-0)
[12:11]	00b – Write-Only	Read/Write Enable (11 – Write 01 – Read, x0 – Reserved)
[10:0]	000000000b – Write-Only	Reserved

Figure 46-1. Serial Test Mode Entry for Write

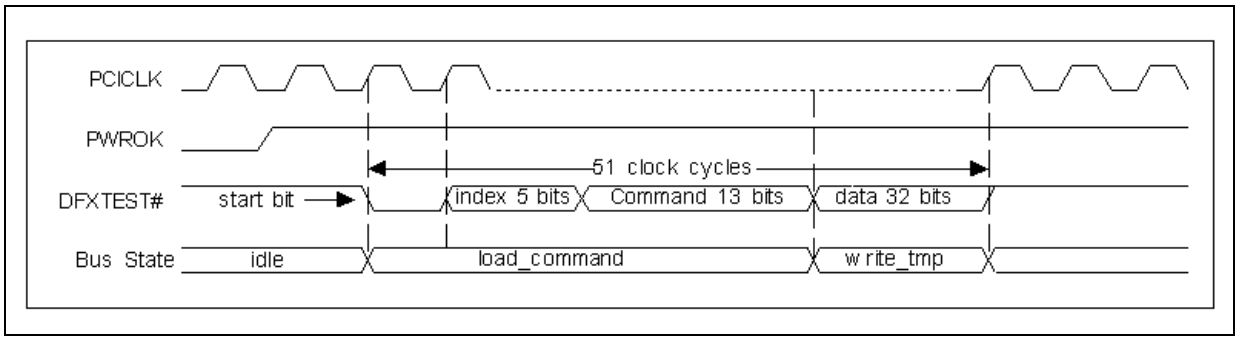
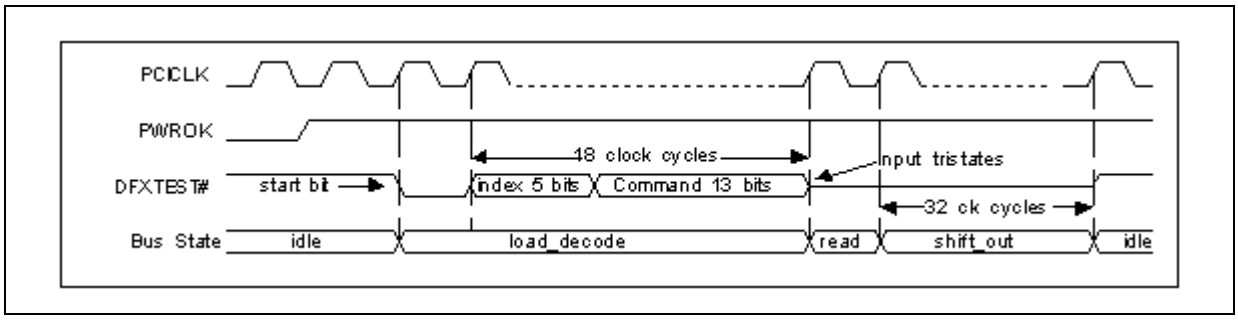


Figure 46-2. Serial Test Mode Entry for Read



46.2.2 Test Mode Registers

46.2.2.1 TEST0 - Test Control Register 0

Table 46-2. Test Control Register 0

Description:					
View: STM	**Base Address:** NA			**Offset Start:** NA **Offset End:** NA	
Size: 32 bit	**Default:** 00010000100000000000000000000000			**Power Well:** Core	
Bit Range	**Bit Acronym**	**Bit Description**	**Sticky**	**Bit Reset Value**	**Bit Access**
31	Reserved	Reserved		0b	RW
30	Reserved	Reserved		0b	RW
29	Reserved	Reserved			RW
28	Reserved	Reserved		0b	RW
27	Reserved	Reserved		1b	RW
26	Reserved	Reserved		0b	RW
25 :24	Reserved	Reserved		00b	RW
23	Reserved	Reserved.		0b	RW
22	Reserved	Reserved		1b	RW
21 :19	Reserved	Reserved		000b	RW
18	Reserved	Reserved		0b	RW
17	Reserved	Reserved		0b	RW
16	Reserved	Reserved		0b	RW
15	Reserved	Reserved		0b	RW
14 :13	Reserved	Reserved		00b	RW
12	Reserved	Reserved		0b	RW
11	Reserved	Reserved		0b	RW
10	Reserved	Reserved		0b	RW
09	Reserved	Reserved		0b	RW
08	XOR	XOR mode enable. Refer to Section 46.2.3 for more information.	Y	0b	RW
07	Reserved	Reserved		0b	RW
06	Reserved	Reserved		0b	RW
05 :04	Reserved	Reserved		00b	RW
03 :01	Reserved	Reserved		000b	WO
00	Reserved	Reserved		0b	RW

46.2.3 XOR Chains

XOR Chains are used to validate connectivity of high speed IO pins when JTAG control is unavailable. This technique chains together pins in a serial fashion, with an XOR gate between each pin, effectively creating a single multi-input XOR gate. By toggling the pins individually or in combination, the connectivity of the pins at the platform level, and the functionality of the I/O driver itself can be validated.

There is only one XOR chain in to cover the AFEs of PCI express, SATA and USB. This mode can be entered by setting bit 8 of test control register 0.

Table 46-3. XOR Chains

XOR Chain Order	Comments
PEA0_Rn[6]	First pin / input pin to XOR chain
PEA0_Rp[6]	
PEA0_Tp[6]	
PEA0_Tn[6]	
PEA0_Rn[7]	
PEA0_Rp[7]	
PEA0_Tp[7]	
PEA0_Tn[7]	
PEA0_Rn[4]	
PEA0_Rp[4]	
PEA0_Tp[4]	
PEA0_Tn[4]	
PEA0_Rn[5]	
PEA0_Rp[5]	
PEA0_Tp[5]	
PEA0_Tn[5]	
PEA_RCOMPO	
PEA_ICOMPO	
PEA0_Rn[2]	
PEA0_Rp[2]	
PEA0_Tp[2]	
PEA0_Tn[2]	
PEA0_Rn[3]	
PEA0_Rp[3]	
PEA0_Tp[3]	
PEA0_Tn[3]	
PEA0_Rn[0]	
PEA0_Rp[0]	
PEA0_Tp[0]	
PEA0_Tn[0]	
PEA0_Rn[1]	
PEA0_Rp[1]	

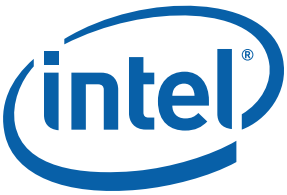
Table 46-3. XOR Chains

PEA0_Tp[1]	
PEA0_Tn[1]	
SATA_RBIAS	
SATA_RBIASSENSE	
SATA_RXn0	
SATA_RXp0	
SATA_TXn0	
SATA_TXp0	
SATA_RXn1	
SATA_RXp1	
SATA_TXn1	
SATA_TXp1	
USBn0	
USBp0	
USBn1	
USBp1	
RI_N	XOR Chain Output pin

These signals listed below enable the XOR test mode for PCI express AFE and need to be connected to dt_xor signal:

- input_comp_oxoren
- input_iol0_oxoren0
- input_iol0_oxoren1
- input_iol1_oxoren0
- input_iol1_oxoren
- input_iou0_oxoren0
- input_iou0_oxoren1
- input_iou1_oxoren0
- input_iou1_oxoren1

§ §



Technical Specifications, Volume 6 of 6

Order Number: 320066-003US

47.0 SKUs, Power Savings and Pre-Boot Firmware

47.1 Overview

This section provides an overview of the various EP80579 SKUs. The SKUs are designed to optimize cost, performance, temperature environment (industial temperature support) and power consumption. Several options allow designers to further reduce power consumption by disabling functional units that may not be utilized by the target application.

47.2 SKUs, Strap Options and Pre-Boot Firmware Programmable Configuration Modes

47.2.1 SKU Features

Table 47-1 outlines the various SKUs available.

Table 47-1. Base Features of EP80579 SKUs

SKU Overview								
Application Target	**Intel® EP80579 Integrated Processor**				**Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology**			
Part Number	**NU80579 EZ600C**	**NU80579 EZ600CT**	**NU80579 EZ004C**	**NU80579 EZ009C**	**NU80579 EB600C**	**NU80579 ED004C**	**NU80579 ED004CT**	**NU80579 ED009C**
SKU ID	**2**	**8**	**4**	**6**	**1**	**3**	**7**	**5**
Processor Frequency (MHz)	600	600	1066	1200/1066	600	1066	1066	1200/1066
Industrial Temp	No	Yes	No	No	No	No	Yes	No
ASU‡	No	No	No	No	Yes	Yes	Yes	Yes
SSU‡	No	No	No	No	Yes	Yes	Yes	Yes
3x TDM‡	No	No	No	No	12 T1/E1	12 T1/E1	12 T1/E1	12 T1/E1
1x DDR2 (MT/s)	400/533/ 667	400/533/ 667	400/533/ 667/800	400/533/ 667/800	400/533/ 667	400/533/ 667/800	400/533/ 667/800	400/533/ 667/800
2x SATA	Gen1, Gen2							
PCI-Ex (root)	1x8, 2x4, or 2x1							
3x GE (10/100/100)	RGMII/RMII							
2x USB 1.1 or 2.0	Yes							
SPI†	Yes							
LPC	Yes							
2x UART	Yes							
2x SMBus	Yes							
WDT	Yes							
RTC	Yes							
36x GPIO	Yes							
2x CAN	Yes							
IEEE 1588-2008 Hardware Assist	Yes							
Local Exp Bus	Yes							
SSP	Yes							

† Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.
‡ Feature must be enabled with the EP80579 software. Refer to the EP80579 software documentation.

47.2.2 DDR2 Frequencies Supported by the EP80579

Supported DDR2 frequencies are based on the SKU and the Tolapai operating frequency. Table 47-2 lists the supported DDR2 frequencies. The supported frequencies are defined as frequency blocks. The DDR2 interface is defined as a function of the SKUs shown in Table 47-1, “Base Features of EP80579 SKUs”.

Table 47-2. IA-32 Core Internal Bus and DDR2 Frequencies

IA-32 Core [MHz]	Internal Quad Pump FSB [MHz]	DDR2 Supported Frequencies (MHz)
1,200	533	400, 533, 667, 800
1,066	533	400, 533, 667, 800
600	400	400, 533, 667

47.2.3 Strap Options (Platform-based Configuration)

The Tolapai provides an external pull-down strap pin (V_SEL) that allows the platform to select IA-32 core frequency. The value of V_SEL determines the operating point (frequency and voltage) of the IA-32 core in conjunction with the value of the Tolapai output pin, BSEL.

BSEL is derived from the IA-32 core BSEL output and indicates the fixed FSB frequency. The V_SEL pin is driven by the Tolapai to a default value corresponding to the SKU shown in Table 47-1, “Base Features of EP80579 SKUs” on page 1730, but may be strapped low by the platform to override the SKU value; this causes the IA-32 core to operate at the lowest gear ratio (multiplier applied to the FSB frequency to obtain the IA-32 core frequency) and voltage combination corresponding to the SKU’s FSB frequency. Table 47-3 shows the Tolapai strap options:

Table 47-3. Tolapai Strap Options

SKU ID	Manufacturing Frequency SKU (FSB / Intel Architecture MHz)	BSEL Value[1]	V_SEL Value	Operating Frequency (FSB / IA-32 Core MHz)	IA-32 Core Voltage
5/6	533/1200	1	1	533/1200	1.30V
	533/1200	1	0	533/1066	1.30V
3/4/7	533/1066	1	0	533/1066	1.30V
--	--	0	1	Reserved	--
1/2/8	400/600	0	0	400/600	1.0V

Notes:
1. The BSEL output to the clock generator on the board assumes that ‘0’ indicates 400 MHz FSB and '1' indicates 533 MHz FSB.

47.2.4 Pre-Boot Firmware Programmable SKU Options

Pre-boot firmware programmable SKU options allow certain features to be disabled. This is helpful when downgrading a part for power savings.

- Power Savings: Software (pre-boot firmware) can disable interface device to save power.

Table 47-4 lists the pre-boot firmware programmable features:

Table 47-4. EP80579 Pre-Boot Firmware Programmable Options

Pre-Boot Firmware Programmable Features[1]
Accelerator Services Unit (ASU)
Security Services Unit (SSU)
Gigabit Ethernet (per port)
CAN 0/1
IEEE 1588-2008 Hardware Assist
SSP
USB
PCI Express* (per controller)
SATA (per port)

Notes:
1. Refer to the EP80579 software and the, Intel® EP80579 Integrated Processor product line Firmware Writer's Guide.

§ §

48.0 Package Specifications

This chapter provides EP80579 package specifications.

48.1 Package Introduction

The EP80579 is offered in a 1,088 ball, Flip-Chip Ball Grid Array (FCBGA) package with Lead-free only (RoHS 5/6 compliant) for commercial and industrial applications.

48.2 Functional Signal Definitions

Table 48-1 provides the legend for interpreting the Type field that appears throughout the tables in this section.

Table 48-1. Signal Type Definitions

Symbol	Description
#	Active low signal
I	Input pin only
O	Output pin only
I/O	Pin can be either an input or output
OD	Open Drain pin
PWR	Power pin
GND	Ground pin
Reservedn	Pin must be connected as described, where n is the reserved pin number
NCn	No Connection, where n is the NC pin number

48.3 JTAG Boundary Scan Chain (BSC) and XOR Chain

The EP80579 implements a consistent, IEEE 1149.1 compliant JTAG Boundary Scan Chain (BSC) for most interfaces (See Section 48.4, “Signal Pin Descriptions” for the interfaces and signals) enabling a low-cost manufacturing test for boards. The interfaces include the IICH interfaces. The BSC cell is defined in the Boundary Scan Description Language (BSDL) documentation and the boundary scan is initiated by the JTAG entry in the IMCH side.

Three high-speed interfaces, PCI Express*, SATA and USB, are not implemented in the JTAG boundary scan. These interfaces are in an XOR Chain to validate connectivity of high speed IO pins when JTAG control is unavailable (See Table 48-2, ”XOR Chain Elements” and Section 48.4, “Signal Pin Descriptions” for the interfaces and signals). XOR chain mode is initiated by the serial test mode entry of the IICH side. The XOR chain provides the sequence of elements that include first element and XOR output (last element) to allow the reader to know the chain starting and ending points. This

technique chains together pins in a serial fashion, with an XOR gate between each pin, effectively creating a single multi-input XOR gate. By toggling the pins individually or in combination, the connectivity of the pins are validated at the output pin.

Note: A test point must be placed on the board for each of these XOR pins.

Table 48-2. XOR Chain Elements

XOR chain elements	XOR chain elements	XOR chain elements
PEA0_Rn[6] -> First Elements	PEA_RCOMPO	PEA0_Tn[1]
PEA0_Rp[6]	PEA_ICOMPO	SATA_RBIASP
PEA0_Tp[6]	PEA0_Rn[2]	SATA_RBIASN
PEA0_Tn[6]	PEA0_Rp[2]	SATA_RXn0
PEA0_Rn[7]	PEA0_Tp[2]	SATA_RXp0
PEA0_Rp[7]	PEA0_Tn[2]	SATA_TXn0
PEA0_Tp[7]	PEA0_Rn[3]	SATA_TXp0
PEA0_Tn[7]	PEA0_Rp[3]	SATA_RXn1
PEA0_Rn[4]	PEA0_Tp[3]	SATA_RXp1
PEA0_Rp[4]	PEA0_Tn[3]	SATA_TXn1
PEA0_Tp[4]	PEA0_Rn[0]	SATA_TXp1
PEA0_Tn[4]	PEA0_Rp[0]	USBn0
PEA0_Rn[5]	PEA0_Tp[0]	USBp0
PEA0_Rp[5]	PEA0_Tn[0]	USBn1
PEA0_Tp[5]	PEA0_Rn[1]	USBp1
PEA0_Tn[5]	PEA0_Rp[1]	RI# -> XOR Output
PEA_ICOMPI	PEA0_Tp[1]	

48.4 Signal Pin Descriptions

This section provides I/O signal definitions. The signals are categorized into one of several blocks:

- IA-32 core
- IMCH
- IICH
- AIOC
- Miscellaneous
- Power

In each block, there are specific functional group interfaces. Table 48-3 summarizes the signal pin tables that appear throughout this section.

Table 48-3. Signal Pin Description References

Block	Interface	Table Reference
IA-32 Core	Thermal Diode	Table 48-4
	Global Clock CRU	Table 48-5
	Sideband Miscellaneous Signals	Table 48-6
IMCH	IMCH Clock/Reset	Table 48-7
	DDR2 Interface	Table 48-8
	PCI Express* Interface	Table 48-9
IICH	Real Time Clock Interface	Table 48-10
	General Purpose I/O Signals (GPIO) and Interrupts	Table 48-11 and Table 48-12
	Serial Peripheral Interface (SPI)	Table 48-13
	Low Pin Count (LPC) Interface	Table 48-14
	SMBus Interface	Table 48-15
	UART Interface	Table 48-16
	Serial ATA (SATA) Interface	Table 48-17
	Universal Serial Bus (USB) Interface	Table 48-18
	Power Management Interface	Table 48-19
	IICH Miscellaneous Signals	Table 48-20
AIOC	Controller Area Network Bus	Table 48-21
	Gigabit Ethernet (GbE) Interface	Table 48-22
	Time Division Multiplexing (TDM) Interface	Table 48-23
	Local Expansion Bus (LEB) Interface	Table 48-24
	Synchronous Serial Port (SSP) Interface	Table 48-25
	IEEE 1588-2008 Hardware Assist Interface	Table 48-26
Misc.	JTAG	Table 48-27
	Miscellaneous Signals	Table 48-28
	Reserved	Table 48-29
	No Connect	Table 48-30
Power	Power and Ground	Table 48-31

The tables in this section list the signal pins associated with each interface and signal group on the EP80579. Some of the pins provide either normal mode or alternate mode. The Signal column provides the name of the signal that is associated with a pin in a particular mode.

48.4.1 IA-32 Core

48.4.1.1 Thermal Diode

Table 48-4. IA-32 Core Thermal Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
THERMDC	Analog	O	1			IA-32 core Thermal Diode: Thermal Diode Cathode (Input).
THERMDA	Analog	I	1			IA-32 core Thermal Diode: Thermal Diode Anode (output).
TOTAL			2			

48.4.1.2 Global Clock CRU

Table 48-5. Global Clock and Reset (CRU) Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
CLKP100	LVDS	I	1			CRU Clock (Differential) Positive side of the 100Mhz clock input.
CLKN100	LVDS	I	1			CRU Clock (Differential): Negative side of the 100Mhz clock input.
TOTAL			2			

48.4.1.3 Sideband Miscellaneous Signals

Table 48-6. Sideband Miscellaneous Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
CPUSLP_OUT#	LVTTL,3.3V	O	1		BSC	CPU Sleep: This EP80579 output signal is made visible to the platform for debug purposes only. This internal EP80579 signal places the processor into a state that saves substantial power compared to the Stop-Grant state. When EP80579 is in this state, it will not recognize snoops or interrupts. The processor will recognize only assertion of the CPURST# signal, and deassertion of CPUSLP_OUT# while in Sleep state. If CPUSLP_OUT# is deasserted, the processor exits Sleep state and returns to Stop-Grant state, restarting its internal clock signals to the bus and processor core units.
INIT33V_OUT#	LVTTL,3.3V	O	1		BSC	CPU Initialization: This 3.3V EP80579 output signal is made visible to the platform to reset the Firmware Hub. Internal EP80579 signal that when asserted, resets integer registers inside the processor without affecting its internal caches or floating-point registers. The processor then begins execution at the power-on Reset vector configured during power-on configuration. The processor continues to handle snoop requests during INIT33V_OUT# assertion. INIT33V_OUT# is an asynchronous signal. However, to ensure recognition of this signal following an Input/Output Write instruction, it must be valid along with the TRDY# assertion of the corresponding Input/ Output Write bus transaction.
NMI	LVTTL,3.3V	I/O	1		BSC	Non-Maskable Interrupt: NMI is used to force a non-maskable interrupt to the processor, if configured. The processor detects an NMI as a rising edge on NMI. NMI is reset by setting the corresponding NMI source enable/disable bit in the NMI Status and Control Register (I/O Register 61h), except when an external platform agent is driving the NMI pin.
SMI_OUT#	LVTTL,3.3V	O	1		BSC	System Management Interrupt: This EP80579 output signal is made visible to the platform for debug purposes only. This internal EP80579 signal is active low output synchronous to PCICLK that is asserted in response to one of many enabled hardware or software events. On accepting a System Management Interrupt, the processor saves the current state and enter System Management mode (SMM). An SMI Acknowledge transaction is issued, and the processor begins program execution from the SMM handler.

Table 48-6. Sideband Miscellaneous Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
STPCLK_OUT#	LVTTL,3.3V	O	1		BSC	Stop Clock Request: This EP80579 output signal is made visible to the platform for debug purposes only. This internal EP80579 signal is an active-low output synchronous to PCICLK that is asserted in response to one of many hardware or software events. When the processor samples STPCLK_OUT# asserted, it causes the processor to enter a low power Stop-Grant state. The processor issues a Stop-Grant Acknowledge transaction, and stops providing internal clock signals to all processor core units except the CPU FSB and CPU APIC units. The processor continues to snoop bus transactions and service interrupts while in Stop-Grant state. When STPCLK_OUT# is deasserted, the processor restarts its internal clock to all units and resumes execution. The assertion of STPCLK_OUT# has no effect on the bus clock; STPCLK_OUT# is an asynchronous CPU input generated by the EP80579 IICH.
RCIN#	LVTTL,3.3V	I	1		BSC	Keyboard Controller Reset Processor: The keyboard controller can generate INIT_N to the processor. This saves the external OR gate of other sources of INIT_N. When the EP80579 detects the assertion of this signal, INIT_N is generated for 16 PCICLK clocks. The EP80579 will ignore RCIN# assertion during transitions to the S3, S4 and S5 states.
A20GATE	LVTTL,3.3V	I	1		BSC	A20 Gate: A signal from the keyboard controller. Acts as an alternative method to force the A20M_N signal active. Saves the external OR gate needed with various other chipsets.
CPURST#	LVTTL,3.3V	O	1		BSC	Processor Bus Reset: The IMCH asserts CPURST# while RSTIN# is asserted and for approximately 1 ms after RSTIN# is deasserted. The CPURST# allows the processors to begin execution in a known state. This EP80579 signal is for the use by the debug tool purpose.
CPUPWRGD_OUT	LVTTL,3.3V	OD O	1	10K Up	BSC	CPUPWRGD is an open drain signal, this signal requires an external pull-up resistor. This EP80579 output signal is made externally visible to the platform for debug purposes only. CPUPWRGD monitors an internal EP80579 signal connected directly from the IICH to the processor and represents a logical AND of PWROK and VRMPWRGD signals. This signal also must be driven high throughout Boundary Scan operation.
IERR#	LVTTL,3.3V	O	1		BSC	IERR (Internal Error) is asserted by a processor as the result of an internal error. Assertion of IERR is usually accompanied by a SHUTDOWN transaction on the FSB. This transaction may optionally be converted to an external error signal (e.g., NMI) by the EP80579. The processor will keep IERR asserted until the assertion of INIT33V_OUT# or the EP80579 is reset using SYS_RESET#. For termination requirements please refer to the platform design guides.
TOTAL			10			

48.4.2 Integrated Memory Controller Hub (IMCH)

48.4.2.1 IMCH Reset

Table 48-7. IMCH Reset Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
RSTIN#	LVTTL,3.3V	I	1		BSC	IMCH Reset Input: This input must be connected to PLTRST#.
PWRGD	LVTTL,3.3V	I	1		BSC	IMCH Power Good: Asynchronously resets the entire IMCH component, including “sticky” bits. Driven by platform logic to indicate all board power supplies are valid.
TOTAL			2			

48.4.2.2 DDR2 SDRAM

Table 48-8. DDR2 Interface Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
DDR_CK[5:0]	SSTL_18	O	6		BSC	DDR Channel Command Clock (Differential): The positive side of the command clocks used by the DDR DRAMs to latch the DDR_A[14:0], DDR_BA[2:0], DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_CKE[1:0], and DDR_CS[1:0]# signals.
DDR_CK[5:0]#	SSTL_18	O	6		BSC	DDR Channel Command Clock (Differential): The negative side of the differential command clock (see DDR_CK[5:0]).
DDR_CS[1:0]#	SSTL_18	O	2		BSC	DDR Channel Chip Select: Used to indicate to which DRAM device cycles are targeted.
DDR_RAS#	SSTL_18	O	1		BSC	DDR Channel Row Address Strobe: Used to indicate a valid row address and open a row
DDR_CAS#	SSTL_18	O	1		BSC	DDR Channel Column Address Strobe: Used to indicate a valid column address and initiate a transaction.
DDR_WE#	SSTL_18	O	1		BSC	DDR Channel Write Enable: Used to indicate a write cycle.
DDR_DM[8:0]	SSTL_18	O	9		BSC	DDR Channel Data Mask: Data mask for write data.
DDR_BA[2:0]	SSTL_18	O	3		BSC	DDR Channel Bank Address: The DDR bank address signals. These signals are outputs of the IMCH and select which bank within a row is selected.
DDR_A[14:0]	SSTL_18	O	15		BSC	DDR Channel Memory Address: The DDR memory address signals.
DDR_DQ[63:0]	SSTL_18	I/O	64		BSC	DDR Channel Data Bus: The DDR data bus provides the data to/from the DRAM devices.
DDR_ECC[7:0]	SSTL_18	I/O	8		BSC	DDR Channel ECC Bits: ECC bits for the data on the interface.

Table 48-8. DDR2 Interface Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
DDR_DQS[8:0]	SSTL_18	I/O	9		BSC	DDR Channel Data Strobes (Differential): The positive side of the data strobe. Each data strobe is used to strobe a set of four or eight data signals (depending on whether x4 or x8 DRAM devices are used).
DDR_DQS[8:0]#	SSTL_18	I/O	9		BSC	DDR Channel Data Strobes (Differential): The negative side of the data strobe (see D_DQS[8:0]).
DDR_CKE[1:0]	SSTL_18	O	2		BSC	DDR Channel Clock Enable: Independent per-DIMM-slot clock enables used by the controller during the initialization sequence.
DDR_ODT0	SSTL_18	I/O	1		BSC	DDR Compensation: On-die termination configuration.
DDR_ODT1	SSTL_18	I/O	1		BSC	DDR Compensation: On-die termination configuration.
DDR_RCOMPX	Analog	I	1			DDR Compensation: CMD/CK/ADD pin slewrate control.
DDR_CRES[2:0]	Analog	I/O	3			DDR Compensation: Resistive compensation I/O's
DDR_DRVCRES	Analog	I/O	1			DDR Compensation
DDR_SLEWCRES	Analog	I/O	1			DDR Compensation: DQ/DQS/DM pin slewrate control.
TOTAL			144			

48.4.2.3 PCI Express*

Table 48-9. PCI Express Interface Signals (Sheet 1 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PEA0_Tp[0]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair. These signals are an output (Tx) from the EP80579's perspective, and must be connected to the input (receive or Rx) signals of the other PCI Express device. The signals for this 1x8 interface can be trained to 2x4 or 2x1 ports. These port configurations map to signals as follows (y is either "n" or "p"): 1x8 Interface Configuration: - PEA0_Ty[7:0] = Link 0 2x4 Interface Configuration: - PEA1_Ty[7:4] = Link 1 - PEA0_Ty[3:0] = Link 0 2x1 Interface Configuration: - PEA1_Ty[4] = Link 1 - PEA0_Ty[0] = Link 0
PEA0_Tn[0]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Rp[0]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair. These signals are an input (Rx) from the EP80579's perspective, and must be connected to the output (transmit or Tx) signals of the other PCI Express device. The signals for this 1x8 interface can be trained to 2x4 or 2x1 ports. These port configurations map to signals as follows (y is either "n" or "p"): 1x8 Interface Configuration: - PEA0_Ry[7:0] = Link 0 2x4 Interface Configuration: - PEA1_Ry[7:4] = Link 1, - PEA0_Ry[3:0] = Link 0 2x1 Interface Configuration: - PEA1_Ry[4] = Link 1, - PEA0_Ry[0] = Link 0
PEA0_Rn[0]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Tp[1]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[1]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])

Table 48-9. PCI Express Interface Signals (Sheet 2 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PEA0_Rp[1]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[1]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA0_Tp[2]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[2]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[2]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[2]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA0_Tp[3]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[3]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[3]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[3]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA0_Tp[4]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[4]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[4]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[4]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).

Table 48-9. PCI Express Interface Signals (Sheet 3 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PEA0_Tp[5]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[5]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[5]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[5]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA0_Tp[6]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[6]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[6]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[6]	LV Diff	I	1		XOR (First element)	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA0_Tp[7]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The positive side of the PCI Express Port A transmit data pair (see PEA0_Tp[0]).
PEA0_Tn[7]	LV Diff	O	1		XOR	PCI Express Interface Port A Transmit Data Pair (Differential): The negative side of the PCI Express Port A transmit data pair (see PEA0_Tn[0])
PEA0_Rp[7]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The positive side of the PCI Express Port A receive data pair (see PEA0_Rp[0]).
PEA0_Rn[7]	LV Diff	I	1		XOR	PCI Express Interface Port A Receive Data Pair (Differential): The negative side of the PCI Express Port A receive data pair (see PEA0_Rn[0]).
PEA_CLKn	LV Diff	I	1			PCI Express Port A Clock (Differential): The negative side of the reference clock input.
PEA_CLKp	LV Diff	I	1			PCI Express Port A Clock (Differential): The positive side of the reference clock input, see PEA_CLKn.
PEA_RCOMPO	Analog	I	1		XOR	PCI Express Port A Compensation: Used to calibrate the PCI Express high speed serial input/output buffers.

Table 48-9. PCI Express Interface Signals (Sheet 4 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PEA_ICOMPI	Analog	I	1		XOR	PCI Express Port A Compensation: Used to calibrate the PCI Express high speed serial input/output buffers
PEA_ICOMPO	Analog	I	1		XOR	PCI Express Port A Compensation: Used to calibrate the PCI Express high speed serial input/output buffers.
TOTAL			37			

48.4.3 Integrated I/O Controller Hub (IICH)

48.4.3.1 Real Time Clock

Table 48-10. Real Time Clock Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
RTCX1	Analog	I	1			RTC Crystal Input 1: Connected to the 32.768 KHz crystal. If no external crystal is used, then RTCX1 can be driven with the desired clock rate.
RTCX2	Analog	O	1			RTC Crystal Input 2: Connected to the 32.768 KHz crystal. If no external crystal is used, then RTCX2 should be left floating.
RTEST#	LVTTL,3.3V	I	1			RTC Test Enable: Oscillator test enable input. An external RC circuit is required to drive this test input.
TOTAL			3			

48.4.3.2 General Purpose I/O (GPIO) and Interrupts

Table 48-11. General-Purpose IO Signals (Sheet 1 of 5)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GPIO[0]	LVTTL,3.3V	I	1		BSC	General Purpose IO 0: Reside in the Core power well.
GPIO[1]	LVTTL,3.3V	I	1		BSC	General Purpose IO 1: Reside in the Core power well.
GPIO[6]	LVTTL,3.3V	I	1		BSC	General Purpose IO 6: Resides in the Core power well. This pin is not available if SMI mode is enabled.
GPIO[7]	LVTTL,3.3V	I	1		BSC	General Purpose IO 7: Resides in the Core power well.
GPIO[8]	LVTTL,3.3V	I	1		BSC	General Purpose IO 8: Resides in the Suspend power well.

Table 48-11. General-Purpose IO Signals (Sheet 2 of 5)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GPIO[9]	LVTTL,3.3V	I	1		BSC	General Purpose IO 9: Resides in the Suspend power well.
GPIO[10]	LVTTL,3.3V	I	1		BSC	General Purpose IO 10: Resides in the Suspend power well.
GPIO[12]	LVTTL,3.3V	I	1		BSC	General Purpose IO 12: Resides in the Core power well.
GPIO[13]	LVTTL,3.3V	I	1		BSC	General Purpose IO 13: Resides in the Core power well.
GPIO[14]	LVTTL,3.3V	I	1		BSC	General Purpose IO 14: Resides in the Suspend power well.
GPIO[15]	LVTTL,3.3V	I	1		BSC	General Purpose IO 15: Resides in the Suspend power well.
GP16_IRQ24	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 16: This signal can function as either GPIO[16] or IRQ[24]. Resides in the Core power well. Note: Internal pullup is always enabled (50K nominal). A16 Override Strap: This strap selects the treatment of A16 for cycles going to BIOS space (but not feature space) in the FWH. The EP80579 interprets this strap as follows: 0 = IICH does not invert A16. 1 = IICH inverts A16 on some BIOS cycles (default)
GP17_IRQ25	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 17: This signal can function as either GPIO[17] or IRQ[25]. Resides in the Core power well. Note: Internal pullup is always enabled (50K nominal). Boot BIOS Selection Strap: This strap selects the source of the BIOS during boot. THe EP80579 interprets this strap as follows: GP17 GP33 Boot Option 0 0 Boot BIOS from SPI 0 1 Reserved 1 0 Reserved 1 1 (Default) Boot BIOS from LPC ***Note:*** Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.
GP18_IRQ36	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 18: This signal can function as either GPIO[18] or IRQ[36]. Resides in the Core power well. Notes: • Internal pullup is always enabled (50K nominal). • Do not drive this signal low until the CPURST# signal is de-asserted.

Table 48-11. General-Purpose IO Signals (Sheet 3 of 5)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GP19_IRQ37	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 19: This signal can function as either GPIO[19] or IRQ[37]. Resides in the Core power well. Notes: • Internal pullup is always enabled (50K nominal). • Do not drive this signal low until the CPURST# signal is de-asserted.
GP20_IRQ26	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 20: This signal can function as either GPIO[20] or IRQ[26]. Resides in the Core power well.
GP21_IRQ27	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 21: This signal can function as either GPIO[21] or IRQ[27]. Resides in the Core power well. Notes: • Internal pullup is always enabled (50K nominal). • Do not drive this signal low until the CPURST# signal is de-asserted.
GP23_IRQ28	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 23: This signal can function as either GPIO[23] or IRQ[28]. Resides in the Core power well. Notes: • Internal pullup is always enabled (50K nominal). • Do not drive this signal low until the CPURST# signal is de-asserted.
GP24_IRQ29	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 24: This signal can function as either GPIO[24] or IRQ[29]. Resides in the Suspend power well.
GP25_IRQ38	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 25: This signal can function as either GPIO[25] or IRQ[38]. Resides in the Suspend power well.
GP27_IRQ39	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 27: This signal can function as either GPIO[27] or IRQ[39]. Resides in the Suspend power well.
GP28_IRQ30	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 28: This signal can function as either GPIO[28] or IRQ[30]. Resides in the Suspend power well.
GP30_IRQ31	LVTTL,3.3V	I	1	No pullup required in IRQ mode	BSC	General Purpose IO 30: This signal can function as either GPIO[30] or IRQ[31]. Resides in the Core power well.
GP31_IRQ32	LVTTL,3.3V	I	1	No pullup required in IRQ mode	BSC	General Purpose IO 31: This signal can function as either GPIO[31] or IRQ[32]. Resides in the Core power well.

Table 48-11. General-Purpose IO Signals (Sheet 4 of 5)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/Down [Ohms]	BSC/XOR	Signal Description Normal/Alternate Mode
GP33_IRQ33	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 33: This signal can function as either GPIO[33] or IRQ[33]. Resides in the Core power well. Note: Internal pullup is always enabled (50K nominal). Boot BIOS Selection Strap: This strap selects the source of the BIOS during boot. The EP80579 interprets this strap as follows: GP17 GP33 Boot Option 0 0 Boot BIOS from SPI 0 1 Reserved 1 0 Reserved 1 1 (Default) Boot BIOS from LPC ***Note:*** Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.
GP34_IRQ34	LVTTL,3.3V	I/O	1	No pullup required in IRQ mode	BSC	General Purpose IO 34: This signal can function as either GPIO[34] or IRQ[34]. Resides in the Core power well.
GP40_IRQ35	LVTTL,3.3V	I	1	No pullup required in IRQ mode	BSC	General Purpose IO 40: This signal can function as either GPIO[40] or IRQ[35]. Resides in the Core power well.
GPIO[48]	LVTTL,3.3V	O	1		BSC	General Purpose IO 48: Resides in the Core power well. Note: Internal pullup is always enabled (50K nominal). The EP80579 interprets this strap as follows: 0 = Reserved 1 = UART/GPIO mode (default).
GP5_PIRQH#	LVTTL,3.3V	I	1	10K Up (in PIRQ mode)	BSC	PCI Interrupt Requests: General purpose or PCI interrupt pin. In Non-APIC mode, the GP5_PIRQH# signal is fixed routed to IRQ[23]. This signal is 5 V tolerant.
GP4_PIRQG#	LVTTL,3.3V	I	1	10K Up (in PIRQ mode)	BSC	PCI Interrupt Requests: General purpose or PCI interrupt pin. In Non-APIC mode, the GP4_PIRQG# signal is fixed routed to IRQ[22]. This signal is 5 V tolerant.
GP3_PIRQF#	LVTTL,3.3V	I	1	10K Up (in PIRQ mode)	BSC	PCI Interrupt Requests: General purpose or PCI interrupt pin. In Non-APIC mode, the GP3_PIRQF# signal is fixed routed to IRQ[21]. This signal is 5 V tolerant.
GP2_PIRQE#	LVTTL,3.3V	I	1	10K Up (in PIRQ mode)	BSC	PCI Interrupt Requests: General purpose or PCI interrupt pin. In Non-APIC mode, the GP2_PIRQE# signal is fixed routed to IRQ[20]. This signal is 5 V tolerant.

Table 48-11. General-Purpose IO Signals (Sheet 5 of 5)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GP41_LDRQ[1]#	LVTTL,3.3V	I	1		BSC	LPC Serial DMA/Master Request Input Bit 1: Used by LPC devices, such as Super I/O chips, to request DMA or bus master access.This signal is typically connected to external Super I/O device. Note: Internal pullup is always enabled (50K nominal). GP41_LDRQ1# may optionally be used as GPI[41].
GP26_SATA0GP	LVTTL,3.3V	I	1		BSC	Serial ATA Port 0 General Purpose: This input pin can be configured as an interlock switch for SATA Port 0 or as a general purpose input, depending on the platform needs. When used as an interlock switch status indication, this signal must be driven to '0' to indicate that the switch is closed, and to '1' to indicate that the switch is open. If interlock switches are not required, the platform can configure this signal as GPI[26]. Note: GP26_SATA0GP and GP29_SATA1GP must be configured for the same purpose (i.e., either both SATAxGP or both GPIO functionality).
GP29_SATA1GP	LVTTL,3.3V	I	1		BSC	Serial ATA Port 1 General Purpose: Same function as GP26_SATA0GP, except for SATA Port 1. When used as an interlock switch status indication, this signal must be driven to '0' to indicate that the switch is closed, and to '1' to indicate that the switch is open. If interlock switches are not required, the platform can configure this signal as GPI[29]. Note: GP26_SATA0GP and GP29_SATA1GP must be configured for the same purpose (i.e., either both SATAxGP or both GPIO functionality).
GP11_SMBALERT#	LVTTL,3.3V	I	1	10K Up (in SMBALERT mode)	BSC	SMBus Alert: This signal is used to wake the system or generate an SMI. Note that the platform can also elect to use this signal as GPI[11] if SMBALERT functionality is not needed. When this signal is used as SMBALERT#, an external pull-up is required.
TOTAL			36			

Table 48-12. IICH Interrupt Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GP5_PIRQH#						See GPIO interface.
GP4_PIRQG#						See GPIO interface.
GP3_PIRQF#						See GPIO interface.

Table 48-12. IICH Interrupt Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GP2_PIRQE#						See GPIO interface.
SERIRQ	LVTTL,3.3V	I	1	10K Up (in PIRQ mode)	BSC	Serial Interrupt Request: This pin implements the serial interrupt protocol. This signal is not 5V tolerant. It is 3.3 V tolerant.
TOTAL			1			

48.4.3.3 Serial Peripheral Interface (SPI)[1]

Table 48-13. SPI Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/ Alternate Mode
SPI_SCLK	LVTTL,3.3V	O	1		BSC	Serial bit-rate Clock
SPI_CS#	LVTTL,3.3V	O	1		BSC	CS for Slave
SPI_MOSI	LVTTL,3.3V	O	1		BSC	Master data out/Slave In
SPI_MISO	LVTTL,3.3V	I	1		BSC	Master data in/Slave out
TOTAL			4			

48.4.3.4 Low Pin Count (LPC) Interface

Table 48-14. LPC and FWH Interface Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
LAD[3:0]	LVTTL,3.3V	I/O	4		BSC	LPC Multiplexed Command, Address, Data LAD[3:0] may be used as Firmware Hub [3:0] signals.
LFRAME#	LVTTL,3.3V	O	1		BSC	LPC Frame: LFRAME# Indicates the start of an LPC cycle, or an abort. LFRAME# may be used as Firmware Hub [4] signal.
GP41_LDRQ[1]#						See GPIO interface.

1. Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.

Table 48-14. LPC and FWH Interface Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
LDRQ[0]#	LVTTL,3.3V	I	1		BSC	LPC Serial DMA/Master Request Input Bit 0: Used by LPC devices, such as Super I/O chips, to request DMA or bus master access.This signal is typically connected to external Super I/O device.
PCICLK	LVTTL,3.3V	I	1		BSC	LPC clock. PCI clock used for the LPC bus (up to 33Mhz)
TOTAL			7			

48.4.3.5 SMBus

Table 48-15. SMBus Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SMBDATA	LVTTL,3.3V	OD I/O	1	8.2K Up	BSC	SMBus Data: SMBus data signal. An external pull-up is required.
SMBCLK	LVTTL,3.3V	OD I/O	1	8.2K Up	BSC	SMBus Clock: SMBus clock signal. An external pull-up is required.
GP11_SMBALERT#						See GPIO interface.
INTRUDER#	LVTTL,3.3V	I	1			SMBus Intruder Detect: Detects if the system case has been opened. Can be set to disables the system if the box is detected open. This input signal is in the RTC well. This pin's status is readable, so it can be used like a GPI if the Intruder switch is not needed.
SMLINK[1:0]	LVTTL,3.3V	OD I/O	2	8.2K Up	BSC	SMBus System Management Link: SMBus link to optional external system management ASIC or LAN controller. External pull-ups are required. Note that SMLINK[0] corresponds to an SMBus Clock signal, and SMLINK[1] corresponds to an SMBus Data signal.
SMBSDA	LVTTL,3.3V	OD I/O	1	8.2K Up	BSC	SMBus Data: Data signal for the IMCH SMBus interface. An external pull-up is required.
SMBSCL	LVTTL,3.3V	OD I/O	1	8.2K Up	BSC	SMBus Clock: Clock signal for the IMCH SMBus interface. An external pull-up is required.
TOTAL			7			

48.4.3.6 UART Interface

Table 48-16. UART Signals (Sheet 1 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SIU_CTS1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Clear to Send: Active low, this pin indicates that data can be exchanged between UART port 1 and the external interface. This pin has no effect on the transmitter. Note: This pin could be used as Modem Status Input whose condition can be tested by the processor by reading bit 4 (CTS) of the Modem Status register (MSR). Bit 4 is the compliment of the CTS# signal. Bit0 (DCTS) of the MSR indicates whether the CTS# input has changed state since the previous reading of the MSR. When the CTS bit of the MSR changes state an interrupt is generated if the Modem Status Interrupt is enabled.
SIU_CTS2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Clear to Send. Refer to UART Port 1 Clear to Send (SIU_CTS1#) for more information.
SIU_DCD1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Data Carrier Detect: Active low, this pin indicates that data carrier has been detected by the external agent for UART port 1. Note: This pin is Modem Status Input which condition can be tested by the processor by reading bit 7 (DCD) of the Modem Status Register (MSR). Bit 7 is complement of the DCD# signal. Bit 3 (DDCD) of the MSR indicates whether the DCD# input has changed state since the previous reading of the MSR. When the DCD bin of the MSR changes state an interrupt is generated if the Modem Status Interrupt is enabled.
SIU_DCD2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Data Carrier Detect. Refer to UART Port 1 Data Carrier Detect (SIU_DCD1) for more information.
SIU_DSR1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Data Set Ready: Active low, this pin indicates that the external agent is ready to communicate with UART port 1. This pin has no effect on the transmitter. Note: This pin could be used as Modem Status Input whose condition can be tested by the processor by reading bit 5 (DSR) of the Modem status register (MSR). Bit 5 is the complement of the DSR# signal. Bit 1 (DDSR) of the Modem status register (MSR) indicates whether the DSR# input has changed state since the previous reading of the MSR. When the DSR bin of the MSR changes state an interrupt is generated if the Modem Status Interrupt is enabled.
SIU_DSR2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Data Set Ready. Refer to UART Port 1 Data Set Ready for more information.

Table 48-16. UART Signals (Sheet 2 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SIU_DTR1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Data Terminal Ready: When low, this pin informs the modem or data set that the UART port 1 is ready to establish a communication link. The DTR# output signal can be set to an active low by programming the DTR (bit 0) of the Modem Control Register to a logic '1'. A Reset operation sets this signal to its inactive state (logic '1'). LOOP mode operation holds this signal in its inactive state. SIW Configuration Port Address Select Strap: This strap selects the IO address for the SIW configuration port. The EP80579 interprets the strap as follows: 0 = IO Addresses 20Eh and 20Fh 1 = IO Addresses 04Eh and 04Fh (default)
SIU_DTR2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Data Terminal Ready. Refer to UART Port 1 Data Terminal Ready (SIU_DTR1#) for more information. Note: Do not drive this signal low until the CPURST# signal is de-asserted.
SIU_RI1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Ring Indicator: Active low, this pin indicates that a telephone ringing signal has been received by the external agent for UART port 1. Note: This pin is Modem Status Input whose condition can be tested by the processor by reading bit 6 (RI) of the MSR. Bit 6 is the complement of the RI# signal. Bit 2 (TERI) of the MSR indicates whether the RI# input has transitioned back to an inactive state. When the RI bit of the MSR changes from a 1 to 0 an interrupt is generated if the Modem Status Interrupt is enabled.
SIU_RI2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Ring Indicator. Refer to UART Port 1 Ring Indicator (SIU_RI1#) for more information.
SIU_RTS1#	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Request To Send: When low this pin informs the modem or data set that UART port 1 is wants to send data on an established communication link. The RTS# output signal can be set to an active low by programming the RTS (bit 1) of the Modem Control Register to a logic '1'. A Reset operation sets this signal to its inactive state (logic '1'). LOOP mode operation holds this signal in its inactive state. Note: Do not drive this signal low until the CPURST# signal is de-asserted.
SIU_RTS2#	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Request to Send. Refer to UART Port 1 Request to Send (SIU_RTS1#) for more information. Note: Do not drive this signal low until the CPURST# signal is de-asserted.
SIU_RXD1	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Serial Data Input: Serial data input form device pin to the receive port for UART port 1.
SIU_RXD2	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Serial Data Input: Serial data input form device pin to the receive port for UART port 2.

Table 48-16. UART Signals (Sheet 3 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SIU_TXD1	LVTTL,3.3V	I/O	1		BSC	UART Port 1 Serial Data Output: Serial data output to the communication peripheral/modem or data set for UART port 1. Upon reset, the TXD pins will be set to MARKING condition (logic '1' state).
SIU_TXD2	LVTTL,3.3V	I/O	1		BSC	UART Port 2 Serial Data Output: Serial data output to the communication peripheral/modem or data set for UART port 2. Upon reset, the TXD pins will be set to MARKING condition (logic '1' state). GPIO IRQ Strap: This strap selects the interrupt capabilities of some GPIO pins on the EP80579 . The EP80579 interprets this strap as follows: 0 = GPIO IRQ capability enabled. 1 = GPIO IRQ capability disabled (default)
UART_CLK	LVTTL,3.3V	I	1		BSC	UART CLock: Input clock to the SIU. This clock is passed to the baud clock generation logic for the UART in the SIU. The clock can run at either 14.7456, 33, or 48 MHz.
TOTAL			17			

48.4.3.7 Serial ATA (SATA) Interface

Table 48-17. Serial ATA Interface Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SATA_TXp0	LV Diff	O	1		XOR	Serial ATA Interface Port 0 Transmit Data Pair (Differential): The positive side of the port 0 transmit data pair.
SATA_TXn0	LV Diff	O	1		XOR	Serial ATA Interface Port 0 Transmit Data Pair (Differential): The negative side of the port 0 transmit data pair.
SATA_RXp0	LV Diff	I	1		XOR	Serial ATA Interface Port 0 Receive Data Pair (Differential): The positive side of the port 0 receive data pair.
SATA_RXn0	LV Diff	I	1		XOR	Serial ATA Interface Port 0 Receive Data Pair (Differential): The negative side of the port 0 receive data pair.
SATA_CLKREFn	CMOS	I	1			Serial ATA Reference Clock (Differential): The negative side of the 100MHz clock input from the clock generator for the SATA PLL.
SATA_CLKREFp	CMOS	I	1			Serial ATA Reference Clock (Differential): The positive side of the 100MHz clock input from the clock generator for the SATA PLL.
SATA_TXp1	LV Diff	O	1		XOR	Serial ATA Interface Port 1 Transmit Data Pair (Differential): The positive side of the port 1 transmit data pair.

Table 48-17. Serial ATA Interface Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SATA_TXn1	LV Diff	O	1		XOR	Serial ATA Interface Port 1 Transmit Data Pair (Differential): The negative side of the port 1 transmit data pair.
SATA_RXp1	LV Diff	I	1		XOR	Serial ATA Interface Port 1 Receive Data Pair (Differential): The positive side of the port 1 receive data pair.
SATA_RXn1	LV Diff	I	1		XOR	Serial ATA Interface Port 1 Receive Data Pair (Differential): The negative side of the port 1 receive data pair.
SATA_RBIAS	Analog	I	1		XOR	Serial ATA Resistor Bias (Positive): Analog connection point for an external bias resistor.
SATA_RBIAS#	Analog	I	1		XOR	Serial ATA Resistor Bias (Negative): Analog connection point for an external bias resistor.
GP26_SATA0GP						See GPIO interface.
GP29_SATA1GP						See GPIO interface.
SATALED#	LVTTL,3.3V	OD O	1	220 to 10K Ohms	BSC	Serial ATA LED: This is an open-collector/open-drain output signal driven during SATA command activity. It is to be connected to external circuitry that can provide the current to drive a platform LED. When active, the LED is on. When tristated, the LED is off. An external pull-up resistor is required.
TOTAL			13			

48.4.3.8 Universal Serial Bus (USB) Interface

Table 48-18. USB Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
USBp0	LV Diff	I/O	1		XOR	USB Port 0 Signal (Differential): The positive side of the differential pair that is used to transmit Data/Address/Command signals for USB port 0. Note: External resistors are not required on these signals. The chip integrates the 15 K-ohm pull-down and provides an output driver impedance of 45 ohm which requires no external series resistor.
USBn0	LV Diff	I/O	1		XOR	USB Port 0 Signal (Differential): The negative side of the differential pair that is used to transmit Data/Address/Command signals for USB port 0 (see USBp0).
USBp1	LV Diff	I/O	1		XOR	USB Port 1 Signal (Differential): The positive side of the differential pairs that are used to transmit Data/Address/Command signals for USB port 0. Note: External resistors are not required on these signals. The chip integrates the 15 K-ohm pull-down and provides an output driver impedance of 45 ohm which requires no external series resistor.
USBn1	LV Diff	I/O	1		XOR	USB Port 1 Signal (Differential): The negative side of the differential pair that is used to transmit Data/Address/Command signals for USB port 0 (see USBp1).
OC[1]#	LVTTL,3.3V	I	1	10K Up	BSC	USB Port 1 Overcurrent Indicator: This signal sets corresponding bits in the USB controller to indicate that an overcurrent condition has occurred on USB Port 1.
OC[0]#	LVTTL,3.3V	I	1	10K Up	BSC	USB Port 0 Overcurrent Indicator: This signal sets corresponding bits in the USB controller to indicate that an overcurrent condition has occurred on USB Port 0.
USB_RBIASp	Analog	O	1			USB Controller Resistor Bias (Positive): Analog connection point for an external resistor. Used to set transmit currents and internal load resistors.
USB_RBIASn	Analog	I	1			USB Controller Resistor Bias (Negative): Analog connection point for an external resistor. Used to set transmit currents and internal load resistors
CLK48	LVTTL,3.3V	I	1			USB Controller Reference Clock: 48MHz reference clock for the USB controller.
TOTAL			9			

48.4.3.9 Power Management Interface

Table 48-19. Power Management Interface Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PLTRST#	LVTTL,3.3V	O	1		BSC	Platform Reset: The EP80579 asserts PLTRST# to reset devices that reside on the PCI bus. The IICH asserts PLTRST# during power-up and when a hard reset sequence is initiated through the 0CF9h register. PLTRST# is driven inactive a minimum of 1 ms after both PWROK and VRMPWRGD are driven high. PLTRST# is driven for a minimum of 1 ms when initiated through the 0CF9h register. Note: PLTRST# is in the 3.3V VCCPSUS well.
PROCHOT#	LVTTL,3.3V	O	1		BSC	Processor Hot Thermal Alarm: This signal will go active when the processor temperature monitoring sensor detects that the processor has reached its maximum safe operating temperature. This indicates that the processor Thermal Control Circuit has been activated, if enabled.
THRMTRIP#	LVTTL,3.3V	OD I/O	1	10K (maximum) Up	BSC	Thermal Trip: The processor protects itself from catastrophic overheating by use of an internal thermal sensor. This sensor is set well above the normal operating temperature to ensure that there are no false trips. When low, indicates that a thermal trip on behalf of the processor has occurred, and corrective action will be taken (immediately transitions the EP80579 to a S5 state). This is an open-drain signal which optionally allows the platform to force an S5 transition.
SLP_S3#	LVTTL,3.3V	O	1		BSC	S3 Sleep Control: SLP_S3# is for power plane control. This signal shuts off power to all non-critical systems when in S3 (Suspend To RAM), S4 (Suspend to Disk), or S5 (Soft Off) states.
SLP_S4#	LVTTL,3.3V	O	1		BSC	S4 Sleep Control: SLP_S4# is for power plane control. This signal shuts power to all non-critical systems when in the S4 (Suspend to Disk) or S5 (Soft Off) state. Note: This pin must be used to control the DRAM power to use the EP80579 's DRAM power-cycling feature.
SLP_S5#	LVTTL,3.3V	O	1		BSC	S5 Sleep Control: SLP_S5# is for power plane control. This signal is used to shut power off to all non-critical systems when in the S5 (Soft Off) states.
PWROK	LVTTL,3.3V	I	1			Power OK: When asserted, PWROK is an indication that core power has been stable for at least 99ms and PCICLK has been toggling cleanly for at least 1 ms. PWROK can be driven asynchronously. When PWROK is low, PLTRST# is asserted. Note that it is required that the core power has been valid for 99ms prior to PWROK assertion in order to comply with the 100ms PCI 2.3 Specification on PLTRST# deassertion.

Table 48-19. Power Management Interface Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PWRBTN#	LVTTL,3.3V	I	1		BSC	Power Button: Causes SMI or SCI to indicate to system request to go to a sleep state. If already in sleep state, will cause a wake event. If PWRBTN# is pressed for four seconds, will cause unconditional transition (power button override) to the S5 state. Override will occur even if the system is in the S3 or S4 state. This signal has an internal pullup resistor and has an internal 16 ms de-bounce on the input.
RI#	LVTTL,3.3V	I	1		BSC/ XOR output	Ring Indicate: From the modem interface. Can be enabled as a wake event and is preserved during power failures.
SYS_RESET#	LVTTL,3.3V	I	1		BSC	System Reset: This pin forces an internal reset after being debounced. The EP80579 will reset immediately if the SMBus is idle; otherwise, it will wait up to 25 ms ± 2 ms for the SMBus to idle before forcing a reset on the system.
RSMRST#	LVTTL,3.3V	I	1			Resume Well Reset: Used for resetting the resume well. An external RC circuit is required to guarantee that the resume well power is valid prior to RSMRST# going high.
SUS_STAT#	LVTTL,3.3V	O	1		BSC	Suspend Status: This signal is asserted to indicate that the system will be entering a low power state soon. This can be monitored by devices with memory that need to switch from normal refresh to suspend refresh mode. It can also be used by other peripherals as an indication that they must isolate their outputs that may be going to powered-off planes. This signal is called LPCPD# on the LPC I/F.
SUSCLK	LVTTL,3.3V	O	1		BSC	Suspend Clock: Output of the RTC clock generator circuit (32.768 kHz). SUSCLK will have a duty cycle that can be as low as 30% or as high as 70%.
VRMPWRGD	LVTTL,3.3V	I	1		BSC	Voltage Regulator Power Good: This is the processor's VRM Power Good, and will save an external AND gate. This signal is internally ANDed with the ATX power supply's PWROK signal. Traditionally, this AND gate has been external to the chipset.
TOTAL			14			

48.4.3.10 IICH Miscellaneous Signals

Table 48-20. IICH Miscellaneous Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
CLK14	LVTTL,3.3V	I	1		BSC	Timer Oscillator Clock: Used for 8254 timers and HPET (High Precision Event Timer). Runs at 14.31818 MHz. This clock stops (and should be low) during S3 and S5 state. CLK14 must be accurate to within 500ppm over 100usecs (and longer periods) in order to meet HPET accuracy requirements.
PE_HPINTR#	LVTTL,3.3V	I	1	10K to 100K Up	BSC	IMCH PCI Express Hot-plug Controller Interrupt: Input pin to hot-plug controller on PCI Express bus. Not 5V tolerant. Note: Because PCI Express Hot Plug is not supported in the EP80579, this pin must be pulled up to 3.3V through a 10K to 100K pull-up resistor.
BSEL	LVTTL,3.3V	O	1		BSC	IA FSB Frequency Select: The CPU Select (BSEL) encodings are used at power-on to specify the frequency mode of the FSB PLL circuitry. The EP80579 interprets BSEL as follows: 0 = 400MHz FSB 1 = 533MHz FSB BSEL is driven by the on-die CPU based on its configuration.
V_SEL	LVTTL,3.3V	I/OD	1	10K Up	BSC	IA Voltage Select: The voltage select encodings are used in conjunction with BSEL at power-on to specify the operating voltage of the IA CPU. This signal is a 3.3V OD output with an external pullup. The EP80579 interprets V_SEL as follows when BSEL is 0 (400MHz FSB): 0 = 1.00V IA core voltage VCCVC 1 = Reserved and as follows when BSEL is 1 (533MHz FSB): 0 = 1.30V IA core voltage VCCVC 1 = 1.30V IA core voltage VCCVC The voltage supply for these pins must be valid before the Voltage Regulator can supply VCCVC to the processor. The V_SEL pin is needed to support the processor voltage specification variations defined above. The VR must supply the voltage that is requested by the pins.
WDT_TOUT#	LVTTL,3.3V	O	1		BSC	Watchdog Timer Output Signal: The signal is driven low when the main 35-bit down counter reaches zero during the second stage. The WDT_TOUT_CNF bit in the WDT Lock register determines if the output is to change from the previous state if another time out occurs, or WDT_TOUT_N is driven low until the system is reset or power is cycled.
TOTAL			5			

48.4.4 Acceleration and I/O Complex (AIOC)

48.4.4.1 Controller Area Network (CAN) Bus

Table 48-21. Controller Area Network Bus Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/Down [Ohms]	BSC/XOR	Signal Description Normal/Alternate Mode
CN0RXD	LVTTL,3.3V	I	1		BSC	CAN Bus Channel 0: Receive data signal. Connect to the respective RxD of the external driver.
CN0TXD	LVTTL,3.3V	O	1		BSC	CAN Bus Channel 0: Transmit data signal. Connect to the respective TxD of the external driver.
CN0TXEN	LVTTL,3.3V	O	1		BSC	CAN Bus Channel 0: Data enable signal.
CN1RXD	LVTTL,3.3V	I	1		BSC	CAN Bus Channel 1: Receive data signal. Connect to the respective RxD of the external driver.
CN1TXD	LVTTL,3.3V	O	1		BSC	CAN Bus Channel 1: Transmit data signal. Connect to the respective TxD of the external driver.
CN1TXEN	LVTTL,3.3V	O	1		BSC	CAN Bus Channel 1: Data enable signal.
TOTAL			6			

48.4.4.2 Gigabit Ethernet (GbE) Interface

Table 48-22. Gigabit Ethernet Interface Signals (Sheet 1 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GBE0_TxCLK	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 0 Transmit Clock: Transmit clock for GbE port 0. The interpretation of this signal depends on the interface mode that the controller is operating in. For RGMII this signal is the Transmit Clock, it is disabled for RMII.
GBE0_TxCTL	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 0 Transmit Control: Transmit control for GbE port 0. The interpretation of this signal depends on the interface mode that the controller is operating in. For RGMII and RMII, this signal is the Transmit Control and Transmit Enable, respectively.
GBE0_TxDATA[3:0]	2.5VCMOS	O	4		BSC	Gigabit Ethernet Controller Port 0 Transmit Control: Transmit control for GbE port 0. The interpretation of these signals depends on the interface mode that the controller is operating in. For RGMII, GBE0_TxDATA[3:0] - Transmit Data and for RMII, GBE0_TxDATA[3:2] - Disabled GBE0_TxDATA[1:0] - Transmit Data
GBE0_RxCLK	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 0 Receive Clock: Receive clock for GbE port 0. The interpretation of this signal depends on the interface mode that the controller is operating in. For RGMII this signal is the Receive Clock, it is disabled for RMII. The GBEn_RxCLK's can be tied to GND as they are not required in RMII mode – only in RGMII.
GBE0_RxCTL	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 0 Receive Control: Receive control for GbE port 0. The interpretation of this signal depends on the interface mode that the controller is operating in. For RGMII and RMII, this signal is the Receive Control and Carrier Sense/Receive Data Valid, respectively.
GBE0_RxDATA[3:0]	2.5VCMOS	I	4		BSC	Gigabit Ethernet Controller Port 0 Receive Data: Receive data for GbE port 0. The interpretation of these signals depends on the interface mode that the controller is operating in. For RGMII, GBE0_RxDATA[3:0] - Receive Data and for RMII, GBE0_RxDATA[3] - Receive Error GBE0_RxDATA[2] - Disabled GBE0_RxDATA[1:0] - Receive Data

Table 48-22. Gigabit Ethernet Interface Signals (Sheet 2 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GBE1_TxCLK	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 1 Transmit Clock: Transmit clock for GbE port 1. See GBE0_TxCLK.
GBE1_TxCTL	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 1 Transmit Control: Transmit control for GbE port 1. See GBE0_TxCTL.
GBE1_TxDATA[3:0]	2.5VCMOS	O	4		BSC	Gigabit Ethernet Controller Port 1 Transmit Data: Transmit data for GbE port 1. See GBE0_TxDATA[3:0].
GBE1_RxCLK	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 1 Receive Clock: Receive clock for GbE port 1. See GBE0_RxCLK.
GBE1_RxCTL	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 1 Receive Control: Receive control for GbE port 1. See GBE0_RxCTL.
GBE1_RxDATA[3:0]	2.5VCMOS	I	4		BSC	Gigabit Ethernet Controller Port 1 Receive Data: Receive data for GbE port 1. See GBE0_RxDATA[3:0].
GBE2_TxCLK	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 2 Transmit Clock: Transmit clock for GbE port 2. See GBE0_TxCLK.
GBE2_TxCTL	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Port 2 Transmit Control: Transmit control for GbE port 2. See GBE0_TxCTL.
GBE2_TxDATA[3:0]	2.5VCMOS	O	4		BSC	Gigabit Ethernet Controller Port 2 Transmit Data: Transmit data for GbE port 2. See GBE0_TxDATA[3:0].
GBE2_RxCLK	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 2 Receive Clock: Receive clock for GbE port 2. See GBE0_RxCLK.
GBE2_RxCTL	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Port 2 Receive Control: Receive control for GbE port 2. See GBE0_RxCTL.
GBE2_RxDATA[3:0]	2.5VCMOS	I	4		BSC	Gigabit Ethernet Controller Port 2 Receive Data: Receive data for GbE port 2. See GBE0_RxDATA[3:0].
MDC	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller Management Channel Clock: Serial clock for the management channel.
MDIO	2.5VCMOS	I/O	1		BSC	Gigabit Ethernet Controller Management Channel Data: Serial data for the management channel.

Table 48-22. Gigabit Ethernet Interface Signals (Sheet 3 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GBE_REFCLK	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller Reference Clock: 50MHz or 125MHz (depend on mode) reference clock for the GbE Controller. In RGMII mode or mixed mode (RGMII and RMII), the frequency of this clock must be 125MHz regardless of the individual port transmit and receive rates and this clock must always be present. In RMII mode (all ports only), there are two clocks required. The GBE_REFCLK_RMII is required as per the RMII spec. The GBE_REFCLK is also required for the RCOMP controller and the retrieval of the port information from the EEPROM. In RMII mode (all ports only), the GBE_REFCLK can be either 50Mhz or 125Mhz. If both clock are the same frequency, connecting GBE_REFCLK and GBE_REFCLK_RMII to the same source will work fine.
GBE_REFCLK_RMII	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller RMII Reference Clock: 50Mhz reference clock for the GbE Controller in RMII Mode. The frequency of this clock is 50MHz regardless of the individual port transmit / receive rates and must always be present.
GBE_RCOMPP	Analog	I/O	1	50 Down (to VSS)		Gigabit Ethernet Controller Compensation: Off-die reference resistor to which the output driver pull up output impedance is matched (nominally 50 ohms to ground).
GBE_RCOMPN	Analog	I/O	1	50 Up (to VCCSUS25)		Gigabit Ethernet Controller Compensation: Off-die reference resistor to which the output driver pull down output impedance is matched (nominally 50 ohms to VCCSUS25).
EEDI	2.5VCMOS	O	1	10K Up (if AUX_PWR_ PRESENT), otherwise 4.7K Down	BSC	Gigabit Ethernet Controller EEPROM Input Data: LVTTL data output to serial EEPROM. This pin is also sampled at an edge of GBE_AUX_PWR_GOOD to indicate the presence of the auxiliary power supply for GbE0. Gigabit Ethernet Controller aux_pwr Strap: This strap indicates if an auxiliary supply is being used for GbE0. The EP80579 interprets the strap as follows: 0 = No aux_pwr mode (i.e., auxiliary supply is not used). 1 = aux_pwr mode support (i.e., auxiliary supply is used)
EEDO	2.5VCMOS	I	1		BSC	Gigabit Ethernet Controller EEPROM Output Data.
EECS	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller EEPROM Chip Select: chip select to enable the EEPROM device.
EESK	2.5VCMOS	O	1		BSC	Gigabit Ethernet Controller EEPROM Shift Clock: clock for the EEPROM interface, frequency is ~1MHz.

Table 48-22. Gigabit Ethernet Interface Signals (Sheet 4 of 4)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
GBE_PME_WAKE	2.5VCMOS	OD O	1		BSC	Gigabit Ethernet Controller Wake: This active low signal is used by the GbE's to signal that a PME event has occurred. It is the wired OR of all three Gigabit Ethernet Controllers on the EP80579 .
SYS_PWR_OK	2.5VCMOS	I	1			Gigabit Ethernet Controller: This is the PWR_GOOD signal used by GbEs supplied by primary core power. This signal indicates that all chip power supplies are on and stable. This signal should be asserted once core power is stable.
GBE_AUX_PWR_GOOD	2.5VCMOS	I	1			Gigabit Ethernet Controller Power Good: Input signifying that the GbE Port 0 and miscellaneous support IO power supplies are on and stable (this is the PWR_GOOD signal which is pinned out for GbE Port 0 only). When an auxiliary power supply is used for GbE Port 0, this signal should be asserted once that auxiliary power is stable. If Core power is used for GbE Port 0 then this pin should be tied to the core Power OK signal SYS_PWR_OK.
TOTAL			49			

48.4.4.3 Time Division Multiplexing (TDM) Interface

Table 48-23. TDM Interface Signals[a,b]

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
Rx_CLK0	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 0: Receive clock.
Tx_CLK0	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 0: Transmit clock.
Tx_FRAME0	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 0: Transmit frame.
Tx_DATA_OUT0	LVTTL,3.3V	O	1		BSC	TDM Interface Port 0: Transmit data output.
Rx_FRAME0	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 0: Receive frame.
Rx_DATA_IN0	LVTTL,3.3V	I	1		BSC	TDM Interface Port 0: Receive data input.
Rx_CLK1	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 1: Receive clock.
Tx_CLK1	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 1: Transmit clock.
Tx_FRAME1	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 1: Transmit frame.
Tx_DATA_OUT1	LVTTL,3.3V	O	1		BSC	TDM Interface Port 1: Transmit data output.
Rx_FRAME1	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 1: Receive frame.
Rx_DATA_IN1	LVTTL,3.3V	I	1		BSC	TDM Interface Port 1: Receive data input.
Rx_CLK2	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 2: Receive clock.
Tx_CLK2	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 2: Transmit clock.
Tx_FRAME2	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 2: Transmit frame.
Tx_DATA_OUT2	LVTTL,3.3V	O	1		BSC	TDM Interface Port 2: Transmit data output.
Rx_FRAME2	LVTTL,3.3V	I/O	1		BSC	TDM Interface Port 2: Receive frame.
Rx_DATA_IN2	LVTTL,3.3V	I	1		BSC	TDM Interface Port 2: Receive data input.
TOTAL			18			

a. Feature must be enabled with the EP80579 software. Refer to the software for the Intel® EP80579 Integrated Processor product line.
b. Certain SKUs may not contain this feature. For complete information about product features, see Section 47.2.1, “SKU Features” .

48.4.4.4 Local Expansion Bus (LEB) Interface

Table 48-24. Expansion Bus Signals (Sheet 1 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
EX_ALE	LVTTL,3.3V	O	1		BSC	Expansion Bus Address Latch Enable/Advance/ LD#: Address latch enable signal for the Expansion bus. EXALE has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXALE is driven by the EP80579.
EX_ADDR[24:0]	LVTTL,3.3V	IO	25		BSC	Expansion Bus Address: The 25-bit address bus for the Expansion Bus. EXADx has pull-ups enabled when PLL_LOCK is deasserted to allow sampling of configuration bits that select the address space size (see strapping discussion). These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXADx driven by the EP80579. Expansion Bus Memory Size Strap: This strap selects the size of the memory space that the Expansion Bus claims for its MMIO region. Bits 23, 22, and 21 encode the size. The EP80579 interprets the value of the EXAD[23:21] pins as follows: 000b - 0MB (i.e., Expansion bus disabled) 001b - 32MB 010b - 64MB 011b - 128MB 1XXb - 256MB Note: Expansion Bus Memory Size should be configured by strapping or by Pre-boot Firmware to determine LEB_SIZE.
EX_BE[1:0]#	LVTTL,3.3V	IO	2		BSC	Expansion Bus Byte Enable: Byte enables for the 16-bit data bus in the Expansion Bus. EXBEx has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXBEx is driven by EP80579. These signals are active-low.
EX_CLK	LVTTL,3.3V	I	1		BSC	Expansion Bus Clock: EXCLK is always an input. May be driven from a GPIO clock or from an external clock source. Maximum Frequency: 80MHz Minimum Frequency: 33MHz

Table 48-24. Expansion Bus Signals (Sheet 2 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
EX_CS[7:0]#	LVTTL,3.3V	IO	8		BSC	Expansion Bus Target Chip Selects: Chip selects to select Expansion Bus devices. EXCS has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXCS is driven by the EP80579. These signals are active-low.
EX_DATA[15:0]	LVTTL,3.3V	IO	16		BSC	Expansion Bus Data: The 16-bit data bus for the Expansion bus. EXDA has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXDA is driven by the EP80579 during outbound writes, and inbound reads, and when the bus is idle.
EX_IOWAIT#	LVTTL,3.3V	I	1		BSC	Expansion bus Target Wait #. EX_IOWAIT_N is always an input. Reset value is driven from the board. This signal is active-low.
EX_PARITY[1:0]	LVTTL,3.3V	IO	2		BSC	Expansion Bus Parity: Parity bits for the two bytes on the data bus. EXPAR0 = Parity for EXDA[7:0] EXPAR1 = Parity for EXDA[15:8] EXPARx has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted. EXPARx is driven with the same timing as EXDAx.
EX_RD#	LVTTL,3.3V	IO	1		BSC	Expansion Bus Read: Read signal for the Expansion Bus. EXRDN has pull-ups enabled when PLL_LOCK is deasserted. These pull-ups are disabled when PLL_LOCK is asserted and the EP80579 drives the signal based upon grant. EXRDN is driven by the EP80579. This signal is active-low.
EX_RDY[3:0]#	LVTTL,3.3V	I	4		BSC	Expansion Bus HPI Ready: HPI ready signal for the Expansion Bus. EXRDYx are always inputs. EXRDY0 = Ready signal for Chip Select #4 EXRDY1 = Ready signal for Chip Select #5 EXRDY2 = Ready signal for Chip Select #6 EXRDY3 = Ready signal for Chip Select #7 This signal is active-low.
EX_BURST	LVTTL,3.3V	I	1		BSC	Expansion Bus Burst Size: Burst size signal. EX_BURST is an input in normal operation.
EX_WR#	LVTTL,3.3V	IO	1		BSC	Expansion Bus Write: Write signal. This signal is active-low.

Table 48-24. Expansion Bus Signals (Sheet 3 of 3)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
EX_RCOMPP	Analog	IO	1	50 Down		Expansion Bus Compensation: Resistive compensation, controls the drive high strength of EX outputs. Connect to a 50 pulldown resistor.
EX_RCOMPN	Analog	IO	1	50 Up (to VCC3P3)		Expansion Bus Compensation: Resistive compensation, controls the drive low strength of the EX outputs. Connect to a 50 pullup resistor, pulled up to VCC3P3.
TOTAL			65			

48.4.4.5 Synchronous Serial Port (SSP) Interface

Table 48-25. SSP Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
SSP_SCLK	LVTTL,3.3V	O	1		BSC	SSP Interface Clock: Serial bit-rate clock (1.84MHz maximum frequency). This is the bit-rate clock, driven from the SSP port to the peripheral. Normally, it toggles only when data is actively being transmitted.
SSP_SFRM	LVTTL,3.3V	O	1		BSC	SSP Interface Frame Indicator: This is the framing signal and indicates the beginning and the end of a serialized data word.
SSP_TXD	LVTTL,3.3V	O	1		BSC	SSP Interface Transmit Data: This is the transmit (i.e., outbound) serialized data lines. The word length and MSB/LSB orientation are a function of the selected serial data format.
SSP_RXD	LVTTL,3.3V	I	1		BSC	SSP Interface Receive Data: This is the receive (i.e., inbound) serialized data lines. The word length and MSB/LSB orientation are a function of the selected serial data format.
SSP_EXTCLK	LVTTL,3.3V	I	1		BSC	SSP Interface External Clock: This clock can be selected to replace the internally-generated clock that is used to generate the serial bit-rate clock (SSPCLK).
TOTAL			5			

48.4.4.6 IEEE 1588-2008 Hardware Assist Interface

Table 48-26. IEEE 1588-2008 Hardware Assist Interface Signals (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
1588_TESTMODE_DATA	LVTTL,3.3V	O	1		BSC	IEEE 1588-2008 Hardware Assist Test Output: This signal will reflect bits of the internal System Timer depending on the setting of internal control bits in the 1588_Test register.
1588_PPS	LVTTL,3.3V	O	1		BSC	IEEE 1588-2008 Hardware Assist Pulse Per Second Output - This signal is asserted high when a match occurs between the Compare register and lower 32 bits of system time. Clearing of this signal is under firmware control. The register and pin can be used to create a pulse per second event.
ASMSSIG	LVTTL,3.3V	I	1		BSC	IEEE 1588-2008 Hardware Assist Auxiliary Slave Mode Snapshot: An active high level on this input causes a snapshot of system time to be captured in the ASMS register.
AMMSSIG	LVTTL,3.3V	I	1		BSC	IEEE 1588-2008 Hardware Assist Auxiliary Master Mode Snapshot: An active high level on this input causes a snapshot of system time to be captured in the AMMS register.

Table 48-26. IEEE 1588-2008 Hardware Assist Interface Signals (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
1588_RX_SNAP	LVTTL,3.3V	O	1		BSC	IEEE 1588-2008 Hardware Assist Receive Snapshot Taken Output: This signal will pulse high for eight pclks whenever a timestamp has been taken on the receive channel. The receive channel that is monitored is determined by the control bits in the TS_Test Register.
1588_TX_SNAP	LVTTL,3.3V	O	1		BSC	IEEE 1588-2008 Hardware Assist Transmit Snapshot Taken Output: This signal will pulse high for eight pclks whenever a timestamp has been taken on the transmit channel. The transmit channel that is monitored is determined by the control bits in the TS_Test Register.
TOTAL			6			

48.4.5 Miscellaneous

48.4.5.1 JTAG

Table 48-27. JTAG Interface Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
TMS	LVCMOS, 1.2V	I	1			Test Interface: Test Mode Select. Input in normal operation.
TDI	LVCMOS, 1.2V	I	1			Test Interface: Test Data In. Input in normal operation.
TDO	LVCMOS, 1.2V	OD O	1			Test Interface: Test Data Out
TCK	LVCMOS, 1.2V	I	1			Test Interface: Test Clock. Input in normal operation.
TRST#	LVCMOS, 1.2V	I	1			Test Interface: Test Reset. Input in normal operation.
BPM0	SSTL_18	OD I/O	1		BSC	ITP debug interface
BPM1	SSTL_18	OD I/O	1		BSC	ITP debug interface
BPM2	SSTL_18	OD I/O	1		BSC	ITP debug interface
BPM3	SSTL_18	OD I/O	1		BSC	ITP debug interface
BPM3_IN	LVCMOS, 1.2V	I	1		BSC	ITP Debug Interface: IA-32 core breakpoint monitor. Should be a NC (no-connect, left floating) if not used.
BPM4_PRDY_OUT	LVTTL,3.3V	OD O	1	50 Ohm Pullup to 1.2v	BSC	ITP Debug Interface: IA-32 core PRDY. Open drain output.
BPM5_PREQ_IN	LVCMOS, 1.2V	I	1		BSC	ITP Debug Interface: IA-32 core PREQ Should be a NC (no-connect, left floating) if not used.
TOTAL			12			

48.4.5.2 Miscellaneous Signals

Table 48-28. Miscellaneous Signals

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
PME#	LVTTL,3.3V	I/OD	1		BSC	PCI Power Management Event: Driven by PCI peripherals to wake the system from low-power states S3, S4 and S5. It can also cause an SCI from the S0 state. Note that in some cases the EP80579 may drive PME# active (low) due to an internal wake event. It will not drive PME# high (but it may be pulled up using the internal pull-up resistor). PME# is in the 3.3V VCCPSUS power plane and has an internal pull-up resistor.
PCIRST#	LVTTL,3.3V	O	1		BSC	PCI Reset: This is the Secondary parallel PCI Bus reset signal. It is a logical OR of the primary interface PLTRST# signal and the state of the Secondary Bus Reset bit. Note: PCIRST# is in the 3.3V VCCPSUS well.
SPKR	LVTTL,3.3V	O	1		BSC	Speaker: The SPKR signal is the output of counter 2 and is internally "ANDed" with Port 061h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the system speaker. Upon PLTRST#, its output state is 0. This signal has a weak internal pull-down resistor. SPKR is sampled at platform reset as a functional strap. No Reboot Strap: This strap controls the re-boot behavior of timeouts in the TCO timer. The EP80579 interprets the strap as follows: 0 = Reboot on second timeout of TCO timer (default). 1 = No re-boot on second timeout of TCO timer.
TOTAL			3			

48.4.5.3 Reserved

Table 48-29. Reserved Pin List (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
Reserved0			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved1			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved2			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.

Table 48-29. Reserved Pin List (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
Reserved3			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved4			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved5			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved6			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved7			1		BSC	Reserved: This signal must be connected to VSS.
Reserved8			1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved9			1		BSC	Reserved: This signal must be connected to VSS.
Reserved10			1	10K Up	BSC	Reserved: Must have external pull-up to VCCPSUS.
Reserved11			1		BSC	Reserved: This pin must be tied low (VSS) externally on the platform.
Reserved12-14			3	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved15			1		BSC	Reserved: This signal must be connected to VSS.
Reserved16	LVTTL,3.3V	I	1	10K Up	BSC	Reserved: This signal must be connected to a 10K-ohm pull-up to 3.3V.
Reserved17	LV	I/O	1		BSC	Reserved: This signal must be connected to VSS.
Reserved18	LVTTL,3.3V	I	1	10K Down	BSC	Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
Reserved19-20	LVTTL,3.3V	I	2	10K Up	BSC	Reserved: This signal must be connected to a 10K-ohm pull-up to 3.3V.
TOTAL			21			

48.4.5.4 No Connect

Table 48-30. No Connect Pin List (Sheet 1 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
NC_SUS_TWO			1		BSC	No Connect
NC_TWO			1		BSC	No Connect
NC7			1		BSC	No Connect
NC9			1		BSC	No Connect
NC10			1		BSC	No Connect
NC11			1		BSC	No Connect

Table 48-30. No Connect Pin List (Sheet 2 of 2)

Signal Name	IO Type	Direction	Ball Count	External Pull-Up/ Down [Ohms]	BSC/ XOR	Signal Description Normal/Alternate Mode
NC12			1			No connect.
NC13			1			No connect.
NC14			1			No connect.
NC15			1			No connect.
NC16			1			No connect.
NC17			1			No Connect
NC18			1			No Connect
NC19			1			No Connect
NC20			1			No Connect
NC21			1			No Connect
NC22			1			No Connect
NC34			1		BSC	No Connect
NC35			1		BSC	No Connect
NC36			1		BSC	No Connect
NC37			1		BSC	No Connect
NC38			1		BSC	No Connect
NC40-NC47			8		BSC	No Connect
NC48			1		BSC	No Connect
NC50-NC53			4		BSC	No Connect
NC54			1		BSC	No Connect
NC55			1		BSC	No Connect
NC56			1	10K Down		Reserved: This signal must be connected to VSS via 10K-ohm pulldown.
NC57-NC59			3		BSC	NoConnect
TOTAL			41			

48.4.6 Power

Table 48-31. Power and Ground Summary Pin List (Sheet 1 of 2)

Signal Name	Signal Description Normal/Alternate Mode	Ball Count
VCCAUSB12	1.2V power supply for USB interface. Analog Power.	3
VCC	1.2V power supply	85
VCC18	1.8V power supply for DDR circuitry. IO Power.	20
VCCTMP18	1.8V power supply for the Thermal Sensor. This supply can be optionally shut off in S3 sustain mode.	1
VCCUSB12	1.2V power supply for USB interface. Digital Power.	7
VCC1P2_USBSUS	1.2V sustain supply for core logic in USB interface.	1
VCC33	3.3V power supply	15

Table 48-31. Power and Ground Summary Pin List (Sheet 2 of 2)

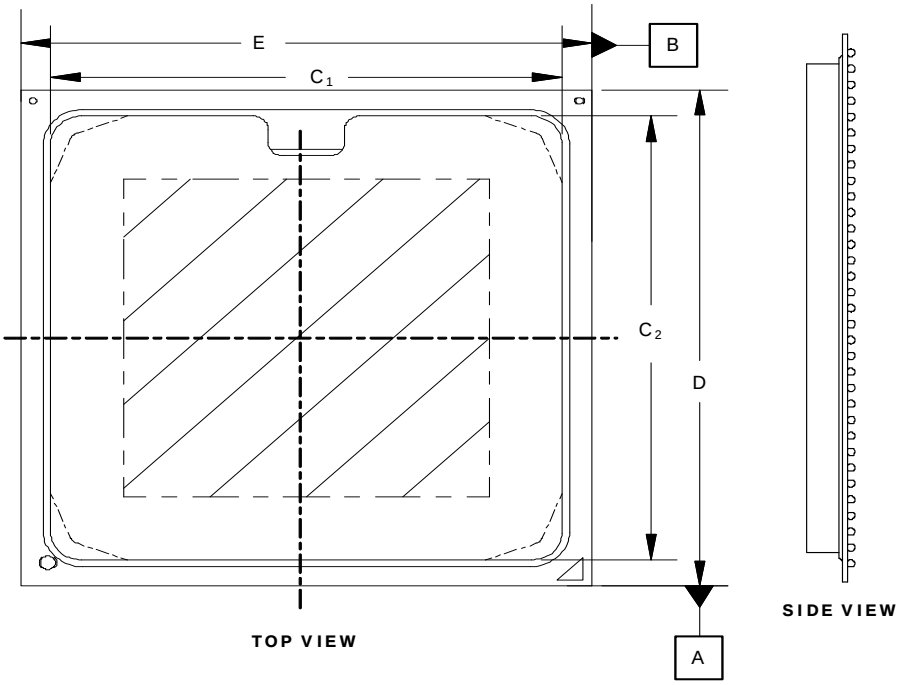
Signal Name	Signal Description Normal/Alternate Mode	Ball Count
VCCGBE33	3.3V power supply for Gigabit Ethernet interfaces.	1
VCCSATA33	3.3V power supply for SATA interface.	1
VCC50	5.0V power supply	7
VCC50_SUS	5.0V power supply for 5.0V tolerance in sustain well	1
VCCA[1]	1.20V power supply for IA-32 core. Minimum voltage is 1.20V-5%, even in low-power applications.	1
VCCA[2]	1.20V power supply for IA-32 core. Minimum voltage is 1.20V-5%, even in low-power applications.	1
VCCABG3P3_USB	3.3V power supply for USB interface. Analog Bandgap Power.	1
VCCABGP033	3.3V power supply for PCI Express circuitry. Analog Bandgap Power.	1
VCCAHPLL	Analog 1.2V supply for CRU circuitry (CRU PLL and REFCLK inputs).	1
VCCAPE	1.2V power supply for PCI Express circuitry. Analog Power.	12
VCCAPE0PLL12	1.2V power supply for PCI Express circuitry. PLL Digital Power.	2
VCCAPLL	1.2V power supply for SATA interface.	1
VCCARX	1.2V power supply for SATA interface. Analog receiver power.	3
VCCASATABG3P3	3.3V power supply for SATA interface. Analog Bandgap power.	1
VCCATX	1.2V power supply for SATA interface. Analog transmitter power.	3
VCCVC	1.30V or 1.00V CPU power supply for IA-32 core. This supply is 1.30V for 1200MHz, 1.30V for 1066MHz, and 1.00V for 600MHz (see V_SEL signal description in the ICH Miscellaneous signals).	1
VCC25	2.5V power supply for Gigabit Ethernet interfaces.	4
VCCSUS25	2.5V sustain power supply for Gigabit Ethernet interfaces.	4
VCCPRTC	3.3V power supply for the RTC (this supply can drop to 2.0V if all other planes are shut off). This power is not expected to be shut off unless the RTC battery is removed or drained.mthellomthelloNote: Implementations must not attempt to clear CMOS by using a jumper to pull VCCPRTC low. Clearing CMOS can be done by using a jumper on RTEST_N or GPI.	1
VCCPSUS	3.3V sustain supply for IO logic	3
VCCGBEPSUS	3.3V sustain supply for IO logic for Gigabit Ethernet interfaces.	1
VCCRPE	1.2V power supply for PCI Express circuitry. Digital Receiver Power.	8
VCCSATA	1.2V power supply for SATA interface.	2
VCCSUS1	1.2V sustain supply for core logic	4
VCCVC	1.30V or 1.00V CPU power supply for IA-32 core. This supply is 1.30V for 1200MHz, 1.30V for 1066MHz, and 1.00V for 600MHz (see V_SEL signal description in the ICH Miscellaneous signals).	45
VSS	Ground	292
VSSA	Analog Ground	1
VTTDDR	0.9V power supply for DDR circuitry. Terminal Voltage.	14
TOTAL		549

48.5 Flip-Chip Ball Grid Array (FCBGA) Package Dimensions

The FCBGA package is shown in the following illustrations:

- Figure 48-1, “FCBGA Package — Top and Side Views” on page 1775

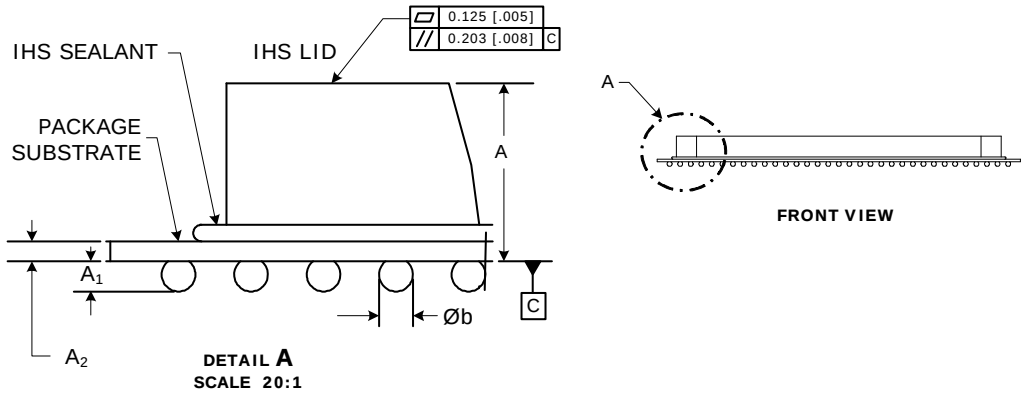
Figure 48-1. FCBGA Package — Top and Side Views



SYMBOL	MILLIMETERS [INCHES]		COMMENTS
	MIN	MAX	
E	37.45 [1.474]	37.55 [1.478]	
D	37.45 [1.474]	37.55 [1.478]	
C_1	33.5 [1.319]		⌖ 1 [.039] C B
C_2	33.5 [1.319]		⌖ 1 [.039] C A
A	3.386 [.133]	3.78 [.149]	
A_1	0.4 [.016]	0.6 [.024]	
A_2	1.085 [.043]	1.255 [.049]	
E_1	34.944 BASIC [1.376]		
D_1	34.944 BASIC [1.376]		
H_1	17.472 BASIC [0.6879]		⌖ Ø0.203 [.008] Ⓛ C A B Ø0.04 [.002] Ⓛ C
H_2	17.472 BASIC [0.6879]		
e	1.092 BASIC [0.043]		
b	0.61 [.024]		

B6452-03

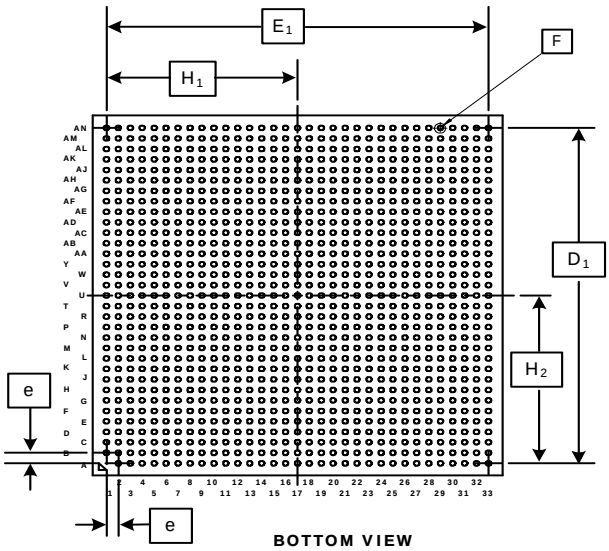
Figure 48-2. FCBGA Package — Front and Detail Views



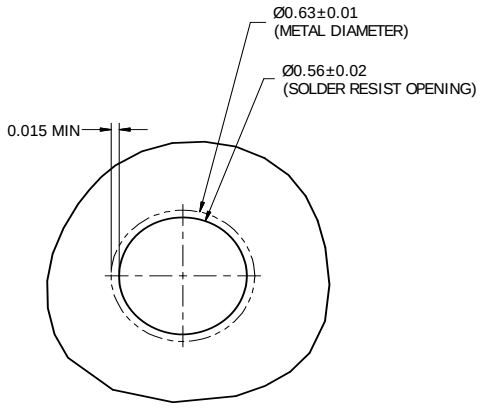
SYMBOL	MILLIMETERS (INCHES)		COMMENTS
	MIN	MAX	
E	37.45 [1.474]	37.55 [1.478]	
D	37.45 [1.474]	37.55 [1.478]	
C_1	33.5 [1.319]		⌖ 1 [.039] C B
C_2	33.5 [1.319]		⌖ 1 [.039] C A
A	3.386 [.133]	3.78 [.149]	
A_1	0.4 [.016]	0.6 [.024]	
A_2	1.085 [.043]	1.255 [.049]	
E_1	34.944 BASIC [1.376]		
D_1	34.944 BASIC [1.376]		
H_1	17.472 BASIC [0.6879]		⌖ Ø0.203 [.008] Ⓛ C A B Ø0.04 [.002] Ⓛ C
H_2	17.472 BASIC [0.6879]		
e	1.092 BASIC [0.043]		
b	0.61 [.024]		

B6453-02

Figure 48-3. FCBGA Package — Bottom View



BOTTOM VIEW



Detail **F**
Multiple Places

SYMBOL	MILLIMETERS [INCHES]		COMMENTS
	MIN	MAX	
E	37.45 [1.474]	37.55 [1.478]	
D	37.45 [1.474]	37.55 [1.478]	
C_1	33.5 [1.319]		⌖ 1 [.039] C B
C_2	33.5 [1.319]		⌖ 1 [.039] C A
A	3.386 [.133]	3.78 [.149]	
A_1	0.4 [.016]	0.6 [.024]	
A_2	1.085 [.043]	1.255 [.049]	
E_1	34.944 BASIC [1.376]		
D_1	34.944 BASIC [1.376]		
H_1	17.472 BASIC [0.6879]		⌖ Ø0.203 [.008] Ⓛ C A B Ø0.04 [.002] Ⓛ C
H_2	17.472 BASIC [0.6879]		
e	1.092 BASIC [0.043]		
F	Solder Resist Opening Ø0.56±0.02 [0.022] See Detail F above		
b	0.61 [0.024]		

B6454-03

48.6 Ball Map Information

This section contains the following ball map information:

Table 48-32. Alphabetical Ball Listing

Ball	Signal
A2	VSS
A3	VSS
A4	GPIO[15]
A5	PLTRST#
A6	SLP_S3#
A7	VSS
A8	NC22
A9	USB_RBIASn
A10	VSS
A11	USBn0
A12	USBn1
A13	VSS
A14	SATA_RXp0
A15	SATA_RXp1
A16	VSS
A17	PEA0_Rn[6]
A18	PEA0_Rp[7]
A19	PEA0_Rn[4]
A20	PEA0_Rp[5]
A21	VSS
A22	PEA_ICOMPO
A23	VSS
A24	PEA0_Rn[3]
A25	PEA0_Rp[2]
A26	PEA0_Rn[1]
A27	PEA0_Rp[0]
A28	VCCAPE
A29	VCCVC
A30	VSS
A31	VSS
A32	VSS
A33	VSS
B1	VSS
B2	VSS
B3	PME#
B4	GPIO[10]
B5	SMLINK[1]
B6	SUS_STAT#
B7	VCCUSB12
B8	NC21
B9	USB_RBIASp
B10	VCCUSB12
B11	USBp0
B12	USBp1
B13	VCCARX
B14	SATA_RXn0
B15	SATA_RXn1
B16	VCCATX
B17	PEA0_Rp[6]
B18	PEA0_Rn[7]
B19	PEA0_Rp[4]
B20	PEA0_Rn[5]
B21	VCCAPE
B22	PEA_ICOMPI
B23	VCCRPE
B24	PEA0_Rp[3]
B25	PEA0_Rn[2]
B26	PEA0_Rp[1]
B27	PEA0_Rn[0]
B28	VSS
B29	Reserved17
B30	VCCVC
B31	CLKN100
B32	VSS
B33	VSS
C1	VSS
C2	VSS
C3	VSS
C4	VSS
C5	SMBCLK
C6	VSS
C7	SLP_S4#
C8	VCCUSB12
C9	VSS
C10	VSS
C11	VCCUSB12
C12	VSS
C13	VCCSATA33
C14	VCCARX
C15	VCCAPLL
C16	VCCATX
C17	VSS
C18	VCCRPE
C19	VCCAPE
C20	VSS
C21	VSS
C22	VCCAPE
C23	VSS
C24	VCCAPE
C25	VSS
C26	VCCRPE
C27	VSS
C28	VCCAPE
C29	VSS
C30	CLKP100
C31	NC35
C32	BPM2
C33	VSS
D1	GP2_PIRQE#
D2	VSS
D3	VSS
D4	GPIO[9]
D5	GPIO[8]
D6	Reserved10
D7	SYS_RESET#
D8	RI#
D9	VSS
D10	VSS
D11	CLK48
D12	OC[1]#
D13	VCCAUSB12
D14	SATA_TXp0
D15	SATA_TXp1
D16	VSS
D17	PEA0_Tn[6]
D18	PEA0_Tp[7]
D19	PEA0_Tn[4]
D20	PEA0_Tp[5]
D21	VCCRPE
D22	PEA_CLKn
D23	VCCRPE
D24	PEA0_Tn[3]
D25	PEA0_Tp[2]

Ball	Signal
D26	PEA0_Tn[1]
D27	PEA0_Tp[0]
D28	NC19
D29	NC54
D30	NC38
D31	DDR_CK[4]
D32	DDR_CK[4]#
D33	VSS
E1	VSS
E2	VSS
E3	VSS
E4	GP11_SMBALERT#
E5	VCC33
E6	VSS
E7	SUSCLK
E8	VSS
E9	VCCUSB12
E10	VSS
E11	VSS
E12	OC[0]#
E13	VSS
E14	SATA_TXn0
E15	SATA_TXn1
E16	VCCATX
E17	PEA0_Tp[6]
E18	PEA0_Tn[7]
E19	PEA0_Tp[4]
E20	PEA0_Tn[5]
E21	PEA_RCOMPO
E22	PEA_CLKp
E23	VCCRPE
E24	PEA0_Tp[3]
E25	PEA0_Tn[2]
E26	PEA0_Tp[1]
E27	PEA0_Tn[0]
E28	NC17
E29	NC34
E30	NC50
E31	DDR_DQ[4]
E32	DDR_DQ[5]
E33	BPM0

Ball	Signal
F1	RTCX1
F2	RTCX2
F3	Reserved11
F4	WDT_TOUT#
F5	VSS
F6	VSS
F7	VSS
F8	VSS
F9	VSS
F10	VCCPSUS
F11	VCCUSB12
F12	NC20
F13	VCCAUSB12
F14	VCCARX
F15	SATA_RBIAS
F16	SATA_RBIAS#
F17	VCCRPE
F18	VCCAPE0PLL12
F19	VCCAPE
F20	VCCAPE0PLL12
F21	VCCAPE
F22	VCCRPE
F23	VCCAPE
F24	VCCA[2]
F25	VCCAPE
F26	THERMDA
F27	THERMDC
F28	NC18
F29	VTTDDR
F30	VSS
F31	DDR_DM[0]
F32	DDR_DQ[0]
F33	DDR_DQ[1]
G1	GP5_PIRQH#
G2	V_SEL
G3	RSMRST#
G4	GP29_SATA1GP
G5	GPIO[14]
G6	SMBDATA
G7	PCIRST#
G8	VSS

Ball	Signal
G9	VSS
G10	VCCUSB12
G11	VCCPSUS
G12	VSS
G13	SATA_CLKREFp
G14	SATA_CLKREFn
G15	VCCABGP033
G16	VCCAPE
G17	VSS
G18	NC15
G19	NC12
G20	VCCAPE
G21	Reserved9
G22	VCCVC
G23	VSS
G24	VCCA[1]
G25	VSS
G26	DDR_CK[5]
G27	DDR_CK[5]#
G28	BPM3
G29	VSS
G30	VCC18
G31	NC40
G32	DDR_DQS[0]
G33	DDR_DQS[0]#
H1	GP3_PIRQF#
H2	GPIO[12]
H3	GP16_IRQ24
H4	VCC33
H5	VSS
H6	VCC33
H7	SMLINK[0]
H8	PWRBTN#
H9	SLP_S5#
H10	VCCSUS1
H11	VSS
H12	VCCABG3P3_USB
H13	VSS
H14	VCCSATA
H15	VCCASATABG3P3
H16	VSS

Ball	Signal
H17	VSS
H18	NC13
H19	NC14
H20	NC16
H21	VCCVC
H22	VSS
H23	VCCVC
H24	VCCVC
H25	VSS
H26	VCCVC
H27	VCCVC
H28	DDR_CS[0]#
H29	VSS
H30	DDR_DQ[7]
H31	DDR_DQ[6]
H32	VSS
H33	DDR_DQ[12]
J1	SERIRQ
J2	VSS
J3	SATALED#
J4	GPIO[7]
J5	GPIO[1]
J6	NC_SUS_TWO
J7	GP25_IRQ38
J8	GP28_IRQ30
J9	VSS
J10	VCCSUS1
J11	VCCPSUS
J12	VCC1P2_USBSUS
J13	VCCAUSB12
J14	VCCSATA
J15	VSS
J16	VCCVC
J17	VSS
J18	VCCVC
J19	VSS
J20	VCCVC
J21	VSS
J22	VCCVC
J23	VSS
J24	VCCVC

Ball	Signal
J25	NC51
J26	VCCVC
J27	DDR_CK[3]#
J28	DDR_CK[3]
J29	VCC18
J30	VSS
J31	DDR_DQ[3]
J32	DDR_DQ[2]
J33	DDR_DQ[13]
K1	LFRAME#
K2	GP18_IRQ36
K3	GP17_IRQ25
K4	VCC50_SUS
K5	VSS
K6	PWROK
K7	GP24_IRQ29
K8	GP27_IRQ39
K9	VCC50
K10	VSS
K11	VCC
K12	VSS
K13	VCC
K14	VSS
K15	VCCVC
K16	VSS
K17	VCCVC
K18	VSS
K19	VCCVC
K20	VSS
K21	VCCVC
K22	VSS
K23	VCCVC
K24	VSS
K25	NC48
K26	NC36
K27	NC37
K28	DDR_CS[1]#
K29	VSS
K30	VCC18
K31	DDR_DQ[8]
K32	DDR_DQ[9]

Ball	Signal
K33	DDR_DM[1]
L1	GP23_IRQ28
L2	LDRQ[0]#
L3	LAD[2]
L4	LAD[3]
L5	LAD[1]
L6	GPIO[0]
L7	RTEST#
L8	INTRUDER#
L9	VSS
L10	VCC
L11	VSS
L12	VCC
L13	VSS
L14	VCC
L15	VSS
L16	VCCVC
L17	VSS
L18	VCCVC
L19	VSS
L20	VCCVC
L21	VSS
L22	VCCVC
L23	VSS
L24	VCCVC
L25	VSS
L26	DDR_ODT1
L27	NC52
L28	VSS
L29	VTTDDR
L30	VSS
L31	DDR_DQS[1]
L32	DDR_DQS[1]#
L33	NC41
M1	RCIN#
M2	SMI_OUT#
M3	VSS
M4	VCC33
M5	VCCPRTC
M6	VSS
M7	CPUPWRGD_OUT

Ball	Signal
M8	BSEL
M9	VCCSUS1
M10	VCCSUS1
M11	VCC
M12	VSS
M13	VCC
M14	VSS
M15	VCC
M16	VSS
M17	VCCVC
M18	VSS
M19	VCCVC
M20	VSS
M21	VCCVC
M22	VSS
M23	VCCVC
M24	VSS
M25	DDR_CKE[1]
M26	BPM1
M27	DDR_DQ[21]
M28	DDR_DQ[20]
M29	VSS
M30	VCC18
M31	DDR_DQ[10]
M32	DDR_DQ[15]
M33	DDR_DQ[14]
N1	CLK14
N2	PCICLK
N3	STPCLK_OUT#
N4	GP21_IRQ27
N5	GP30_IRQ31
N6	LAD[0]
N7	GP4_PIRQG#
N8	GPIO[13]
N9	GPIO[6]
N10	VCC
N11	VSS
N12	VCC
N13	VSS
N14	VCC
N15	VSS

Ball	Signal
N16	VCC
N17	VSS
N18	VCCVC
N19	VSS
N20	VCCVC
N21	VSS
N22	VCCVC
N23	VSS
N24	VCCVC
N25	VCCTMP18
N26	VSS
N27	DDR_DQ[17]
N28	DDR_DQ[16]
N29	VCC18
N30	VSS
N31	DDR_DQ[29]
N32	DDR_DQ[28]
N33	DDR_DQ[11]
P1	GP26_SATA0GP
P2	SPKR
P3	GP40_IRQ35
P4	GP33_IRQ33
P5	CPUSLP_OUT#
P6	GP41_LDRQ[1]#
P7	GP20_IRQ26
P8	GP19_IRQ37
P9	VCC50
P10	VSS
P11	VCC
P12	VSS
P13	VCC
P14	VSS
P15	VCC
P16	VSS
P17	VCC
P18	VSS
P19	VCCVC
P20	VSS
P21	VCCVC
P22	VSS
P23	VCCVC

Ball	Signal
P24	VSS
P25	VCCVC
P26	VSS
P27	NC42
P28	DDR_DM[2]
P29	VSS
P30	VCC18
P31	VSS
P32	DDR_DQ[25]
P33	DDR_DQ[24]
R1	SIU_CTS1#
R2	SIU_TXD1
R3	SIU_RXD1
R4	VSS
R5	NMI
R6	VCC33
R7	VSS
R8	VCC
R9	VCC33
R10	VCC
R11	VSS
R12	VCC
R13	VSS
R14	VCC
R15	VSS
R16	VCC
R17	VSS
R18	VCCVC
R19	VSS
R20	VCCVC
R21	VSS
R22	VCCVC
R23	VSS
R24	VCCVC
R25	VSS
R26	VSS
R27	VCC18
R28	VSS
R29	DDR_DQS[2]
R30	DDR_DQS[2]#
R31	VSS

Ball	Signal
R32	DDR_DM[3]
R33	NC43
T1	SIU_DCD2#
T2	SIU_CTS2#
T3	SIU_TXD2
T4	SIU_RXD2
T5	SIU_DCD1#
T6	PROCHOT#
T7	GP34_IRQ34
T8	NC7
T9	VSS
T10	VSS
T11	VCC
T12	VSS
T13	VCC
T14	VSS
T15	VCC
T16	VSS
T17	VCC
T18	VSS
T19	VCCAHPLL
T20	VSS
T21	VCCVC
T22	VSS
T23	VCCVC
T24	VCCVC
T25	VCC18
T26	VSS
T27	DDR_DQ[31]
T28	DDR_DQ[18]
T29	DDR_DQ[23]
T30	DDR_DQ[22]
T31	DDR_DQ[30]
T32	DDR_DQS[3]#
T33	DDR_DQS[3]
U1	CPURST#
U2	PE_HPINTR#
U3	NC_TWO
U4	THRMTRIP#
U5	GPIO[48]
U6	SIU_RTS1#

Ball	Signal
U7	SIU_DTR1#
U8	UART_CLK
U9	VCC50
U10	VCC
U11	VSS
U12	VCC
U13	VSS
U14	VCC
U15	VSS
U16	VCC
U17	VSS
U18	VCC
U19	VSSA
U20	VCC
U21	VSS
U22	VCC
U23	VCCVC
U24	VCC
U25	VCCVC
U26	VCCVC
U27	DDR_DQ[27]
U28	DDR_DQ[19]
U29	DDR_CRES[2]
U30	DDR_CRES[1]
U31	DDR_CK[2]#
U32	DDR_CK[2]
U33	DDR_DQ[26]
V1	INIT33V_OUT#
V2	A20GATE
V3	VSS
V4	VCC33
V5	SIU_DSR1#
V6	VSS
V7	VRMPWRGD
V8	VSS
V9	NC10
V10	VSS
V11	VCC
V12	VSS
V13	VCC
V14	VSS

Ball	Signal
V15	VCC
V16	VSS
V17	VCC
V18	VSS
V19	VCC
V20	VSS
V21	VCC
V22	VTTDDR
V23	VCC
V24	VTTDDR
V25	VSS
V26	DDR_RCOMPX
V27	DDR_CRES[0]
V28	DDR_SLEWCRES
V29	VTTDDR
V30	DDR_CK[0]
V31	DDR_CK[0]#
V32	DDR_ECC[4]
V33	DDR_DRVCRES
W1	SIU_RI2#
W2	GP31_IRQ32
W3	SIU_RI1#
W4	SIU_DSR2#
W5	BPM5_PREQ_IN
W6	PWRGD
W7	Reserved15
W8	NC9
W9	NC11
W10	VCC
W11	VSS
W12	VCC
W13	VSS
W14	VCC
W15	VSS
W16	VCC
W17	VSS
W18	VCC
W19	VSS
W20	VCC
W21	VSS
W22	VCC

Ball	Signal
W23	VSS
W24	VCC
W25	VSS
W26	DDR_DQ[37]
W27	DDR_DQ[32]
W28	DDR_DQ[33]
W29	VSS
W30	VCC18
W31	DDR_ECC[5]
W32	DDR_CK[1]
W33	DDR_CK[1]#
Y1	SMBSDA
Y2	BPM3_IN
Y3	TCK
Y4	SIU_RTS2#
Y5	SMBSCL
Y6	Reserved16
Y7	TRST#
Y8	Reserved6
Y9	VCC50
Y10	VSS
Y11	VCC
Y12	VSS
Y13	VCC
Y14	VSS
Y15	VCC
Y16	VSS
Y17	VCC
Y18	VSS
Y19	VCC
Y20	VSS
Y21	VCC
Y22	VSS
Y23	VCC
Y24	VSS
Y25	DDR_DQ[36]
Y26	DDR_DM[4]
Y27	DDR_DQS[4]
Y28	DDR_DQS[4]#
Y29	VCC18
Y30	VSS

Ball	Signal
Y31	DDR_DM[8]
Y32	DDR_ECC[0]
Y33	DDR_ECC[1]
AA1	TDI
AA2	SIU_DTR2#
AA3	BPM4_PRDY_OUT
AA4	VSS
AA5	Reserved8
AA6	VCC33
AA7	Reserved7
AA8	VSS
AA9	Tx_DATA_OUT1
AA10	VCC
AA11	VSS
AA12	VCC
AA13	VSS
AA14	VCC
AA15	VSS
AA16	VCC
AA17	VSS
AA18	VCC
AA19	VSS
AA20	VCC
AA21	VSS
AA22	VCC
AA23	VSS
AA24	VCC
AA25	VTTDDR
AA26	NC44
AA27	DDR_DQ[38]
AA28	DDR_DQ[39]
AA29	VSS
AA30	VCC18
AA31	NC53
AA32	DDR_DQS[8]
AA33	DDR_DQS[8]#
AB1	Reserved5
AB2	Reserved3
AB3	Reserved0
AB4	Tx_CLK0
AB5	TMS

Ball	Signal
AB6	Reserved4
AB7	Tx_FRAME0
AB8	Rx_FRAME1
AB9	Reserved18
AB10	VSS
AB11	VCC
AB12	VSS
AB13	VCC
AB14	VSS
AB15	VCC
AB16	VSS
AB17	VCC
AB18	VSS
AB19	VCC
AB20	VSS
AB21	VCC
AB22	VSS
AB23	VCC
AB24	VSS
AB25	VSS
AB26	DDR_DQ[34]
AB27	DDR_DQ[52]
AB28	DDR_DQ[35]
AB29	VTTDDR
AB30	VSS
AB31	DDR_ECC[2]
AB32	DDR_ECC[6]
AB33	DDR_ECC[7]
AC1	Tx_DATA_OUT0
AC2	Rx_FRAME0
AC3	RSTIN#
AC4	IERR#
AC5	Reserved1
AC6	Rx_CLK0
AC7	VSS
AC8	SPI_MOSI
AC9	VCC50
AC10	VCC
AC11	VSS
AC12	VCC
AC13	VSS

Ball	Signal
AC14	VCC
AC15	VSS
AC16	VCC
AC17	VSS
AC18	VCC
AC19	VSS
AC20	VCC
AC21	VSS
AC22	VCC
AC23	VSS
AC24	VCC
AC25	VTTDDR
AC26	DDR_DQ[49]
AC27	DDR_DQ[48]
AC28	DDR_DQ[53]
AC29	VSS
AC30	VCC18
AC31	DDR_DQ[45]
AC32	DDR_ECC[3]
AC33	DDR_DQ[44]
AD1	Tx_FRAME1
AD2	Rx_CLK1
AD3	VSS
AD4	VCC33
AD5	Tx_CLK2
AD6	Tx_FRAME2
AD7	Rx_DATA_IN2
AD8	CN0TXD
AD9	NC59
AD10	VSS
AD11	VCC
AD12	VSS
AD13	VCC
AD14	VSS
AD15	VCC
AD16	VSS
AD17	VCC
AD18	VSS
AD19	VCC
AD20	VSS
AD21	VCC

Ball	Signal
AD22	VSS
AD23	VCC
AD24	VSS
AD25	VSS
AD26	DDR_DQ[55]
AD27	DDR_DM[6]
AD28	NC46
AD29	VCC18
AD30	VSS
AD31	DDR_DM[5]
AD32	DDR_DQ[40]
AD33	DDR_DQ[41]
AE1	Rx_DATA_IN1
AE2	Tx_DATA_OUT2
AE3	Rx_FRAME2
AE4	Rx_DATA_IN0
AE5	Rx_CLK2
AE6	NC56
AE7	CN0TXEN
AE8	VSS
AE9	VCC33
AE10	VSS
AE11	VCC50
AE12	VSS
AE13	VSS
AE14	VCC50
AE15	Reserved19
AE16	GBE_PME_WAKE
AE17	GBE_RCOMPN
AE18	VCCSUS25
AE19	VCCGBE33
AE20	VCC25
AE21	VSS
AE22	VSS
AE23	VSS
AE24	VTTDDR
AE25	VSS
AE26	DDR_DQ[50]
AE27	DDR_DQS[6]
AE28	DDR_DQS[6]#
AE29	VSS

Ball	Signal
AE30	VCC18
AE31	NC45
AE32	DDR_DQS[5]
AE33	DDR_DQS[5]#
AF1	SPI_MISO
AF2	SPI_CS#
AF3	SPI_SCLK
AF4	Tx_CLK1
AF5	CN0RXD
AF6	VCC33
AF7	SSP_RXD
AF8	EX_ADDR[19]
AF9	EX_ADDR[11]
AF10	EX_ADDR[5]
AF11	EX_CS[5]#
AF12	VSS
AF13	VSS
AF14	EX_DATA[1]
AF15	EX_RDY[1]#
AF16	EESK
AF17	GBE_RCOMPP
AF18	VSS
AF19	GBE1_TxCTL
AF20	GBE1_TXDATA[1]
AF21	GBE2_TXDATA[3]
AF22	GBE2_TXDATA[0]
AF23	VTTDDR
AF24	VSS
AF25	VSS
AF26	DDR_DQ[51]
AF27	DDR_RAS#
AF28	DDR_DQ[54]
AF29	VTTDDR
AF30	VSS
AF31	DDR_DQ[46]
AF32	DDR_DQ[42]
AF33	DDR_DQ[47]
AG1	1588_RX_SNAP
AG2	TDO
AG3	1588_TX_SNAP
AG4	VSS

Ball	Signal
AG5	CN1TXEN
AG6	VSS
AG7	EX_ADDR[17]
AG8	EX_ADDR[13]
AG9	EX_ADDR[7]
AG10	EX_CS[3]#
AG11	EX_RCOMPN
AG12	EX_DATA[13]
AG13	EX_DATA[5]
AG14	NC57
AG15	EX_RDY[3]#
AG16	EEDI
AG17	MDC
AG18	VCCSUS25
AG19	VSS
AG20	VCCGBEPSUS
AG21	GBE2_TxCTL
AG22	GBE2_TXDATA[1]
AG23	VSS
AG24	VSS
AG25	VTTDDR
AG26	DDR_BA[2]
AG27	DDR_A[14]
AG28	VSS
AG29	VSS
AG30	VCC18
AG31	DDR_DQ[61]
AG32	DDR_DQ[43]
AG33	DDR_DQ[60]
AH1	Reserved2
AH2	ASMSSIG
AH3	SSP_EXTCLK
AH4	SSP_SFRM
AH5	VSS
AH6	EX_ADDR[21]
AH7	EX_ADDR[20]
AH8	VSS
AH9	EX_ADDR[1]
AH10	EX_CLK
AH11	VCC33
AH12	EX_DATA[11]

Ball	Signal
AH13	EX_DATA[3]
AH14	VSS
AH15	NC58
AH16	EECS
AH17	MDIO
AH18	GBE0_TXDATA[0]
AH19	GBE1_TxCLK
AH20	GBE1_TXDATA[3]
AH21	GBE2_RXDATA[1]
AH22	GBE2_TXDATA[2]
AH23	VSS
AH24	VSS
AH25	VSS
AH26	DDR_A[12]
AH27	DDR_A[9]
AH28	DDR_A[11]
AH29	VCC18
AH30	VSS
AH31	DDR_DQS[7]#
AH32	DDR_DQ[56]
AH33	DDR_DQ[57]
AJ1	AMMSSIG
AJ2	VSS
AJ3	CN1TXD
AJ4	SSP_SCLK
AJ5	EX_ADDR[23]
AJ6	EX_ADDR[15]
AJ7	VSS
AJ8	VSS
AJ9	EX_ADDR[3]
AJ10	EX_CS[7]#
AJ11	EX_RCOMPP
AJ12	VSS
AJ13	EX_DATA[7]
AJ14	Reserved20
AJ15	EX_RDY[2]#
AJ16	Reserved12
AJ17	SYS_PWR_OK
AJ18	GBE0_TXDATA[3]
AJ19	VCC25
AJ20	VSS

Ball	Signal
AJ21	VCC25
AJ22	GBE2_TxCLK
AJ23	VSS
AJ24	VSS
AJ25	VTTDDR
AJ26	DDR_A[7]
AJ27	DDR_A[8]
AJ28	DDR_A[6]
AJ29	VSS
AJ30	VCC18
AJ31	DDR_DQS[7]
AJ32	DDR_DM[7]
AJ33	NC47
AK1	1588_PPS
AK2	CN1RXD
AK3	SSP_TXD
AK4	EX_ALE
AK5	EX_ADDR[9]
AK6	VSS
AK7	EX_ADDR[12]
AK8	EX_ADDR[2]
AK9	EX_ADDR[0]
AK10	VSS
AK11	EX_DATA[14]
AK12	VSS
AK13	EX_DATA[4]
AK14	EX_RD#
AK15	VSS
AK16	EEDO
AK17	VSS
AK18	VCCSUS25
AK19	GBE0_TXDATA[2]
AK20	GBE0_TXDATA[1]
AK21	GBE1_TXDATA[2]
AK22	GBE1_RXDATA[0]
AK23	GBE1_TXDATA[0]
AK24	VSS
AK25	VCC18
AK26	DDR_A[5]
AK27	DDR_A[4]
AK28	DDR_A[3]

Ball	Signal
AK29	VTTDDR
AK30	VSS
AK31	VSS
AK32	VSS
AK33	VSS
AL1	VSS
AL2	1588_TESTMODE_DATA
AL3	EX_ADDR[24]
AL4	VSS
AL5	VCC33
AL6	EX_ADDR[14]
AL7	EX_BE[1]#
AL8	VSS
AL9	VCC33
AL10	EX_CS[0]#
AL11	EX_DATA[12]
AL12	EX_DATA[9]
AL13	EX_DATA[2]
AL14	EX_PARITY[1]
AL15	VCC33
AL16	Reserved13
AL17	GBE_REFCLK_RMII
AL18	GBE0_RxCLK
AL19	GBE0_TxCLK
AL20	GBE0_TxCTL
AL21	GBE1_RXDATA[3]
AL22	GBE2_RxCLK
AL23	GBE2_RXDATA[2]
AL24	GBE2_RXDATA[0]
AL25	VSS
AL26	NC55
AL27	DDR_A[1]
AL28	DDR_A[2]
AL29	DDR_ODT0
AL30	DDR_DQ[58]
AL31	DDR_DQ[62]
AL32	DDR_DQ[63]
AL33	VSS
AM1	VSS
AM2	VSS
AM3	EX_ADDR[22]

Ball	Signal
AM4	EX_ADDR[18]
AM5	EX_ADDR[10]
AM6	VSS
AM7	EX_ADDR[4]
AM8	EX_CS[6]#
AM9	EX_CS[2]#
AM10	VSS
AM11	EX_DATA[10]
AM12	EX_DATA[6]
AM13	VSS
AM14	EX_PARITY[0]
AM15	EX_RDY[0]#
AM16	Reserved14
AM17	VCCSUS25
AM18	GBE0_RxCTL
AM19	VSS
AM20	GBE0_RXDATA[0]
AM21	GBE1_RxCTL
AM22	VCC25
AM23	VSS
AM24	GBE2_RXDATA[3]
AM25	VCC18
AM26	DDR_A[0]
AM27	DDR_A[10]
AM28	DDR_BA[0]
AM29	DDR_BA[1]
AM30	DDR_DQ[59]
AM31	VCC18
AM32	VSS
AM33	VSS
AN1	VSS
AN2	VSS
AN3	VSS
AN4	EX_ADDR[16]
AN5	EX_ADDR[8]
AN6	EX_ADDR[6]
AN7	EX_BE[0]#
AN8	EX_CS[4]#
AN9	EX_CS[1]#
AN10	EX_DATA[15]
AN11	EX_DATA[8]

Ball	Signal
AN12	EX_DATA[0]
AN13	EX_IOWAIT#
AN14	EX_BURST
AN15	EX_WR#
AN16	GBE_AUX_PWR_GOOD
AN17	GBE_REFCLK
AN18	GBE0_RXDATA[3]
AN19	GBE0_RXDATA[2]
AN20	GBE0_RXDATA[1]
AN21	GBE1_RxCLK
AN22	GBE1_RXDATA[2]
AN23	GBE1_RXDATA[1]
AN24	GBE2_RxCTL
AN25	VSS
AN26	DDR_CAS#
AN27	DDR_A[13]
AN28	DDR_WE#
AN29	DDR_CKE[0]
AN30	VSS
AN31	VSS
AN32	VSS
AN33	VSS

Table 48-33. Alphabetical Signal Listing

Signal	Ball
1588_PPS	AK1
1588_RX_SNAP	AG1
1588_TESTMODE_DATA	AL2
1588_TX_SNAP	AG3
A20GATE	V2
AMMSSIG	AJ1
ASMSSIG	AH2
BPM0	E33
BPM1	M26
BPM2	C32
BPM3	G28
BPM3_IN	Y2
BPM4_PRDY_OUT	AA3
BPM5_PREQ_IN	W5
BSEL	M8
CLK14	N1
CLK48	D11
CLKN100	B31
CLKP100	C30
CN0RXD	AF5
CN0TXD	AD8
CN0TXEN	AE7
CN1RXD	AK2
CN1TXD	AJ3
CN1TXEN	AG5
CPUPWRGD_OUT	M7
CPURST#	U1
CPUSLP_OUT#	P5
DDR_A[0]	AM26
DDR_A[1]	AL27
DDR_A[2]	AL28
DDR_A[3]	AK28
DDR_A[4]	AK27
DDR_A[5]	AK26
DDR_A[6]	AJ28
DDR_A[7]	AJ26
DDR_A[8]	AJ27
DDR_A[9]	AH27
DDR_A[10]	AM27
DDR_A[11]	AH28
DDR_A[12]	AH26

Signal	Ball
DDR_A[13]	AN27
DDR_A[14]	AG27
DDR_BA[0]	AM28
DDR_BA[1]	AM29
DDR_BA[2]	AG26
DDR_CAS#	AN26
DDR_CK[0]	V30
DDR_CK[0]#	V31
DDR_CK[1]	W32
DDR_CK[1]#	W33
DDR_CK[2]	U32
DDR_CK[2]#	U31
DDR_CK[3]	J28
DDR_CK[3]#	J27
DDR_CK[4]	D31
DDR_CK[4]#	D32
DDR_CK[5]	G26
DDR_CK[5]#	G27
DDR_CKE[0]	AN29
DDR_CKE[1]	M25
DDR_CRES[0]	V27
DDR_CRES[1]	U30
DDR_CRES[2]	U29
DDR_CS[0]#	H28
DDR_CS[1]#	K28
DDR_DM[0]	F31
DDR_DM[1]	K33
DDR_DM[2]	P28
DDR_DM[3]	R32
DDR_DM[4]	Y26
DDR_DM[5]	AD31
DDR_DM[6]	AD27
DDR_DM[7]	AJ32
DDR_DM[8]	Y31
DDR_DQ[0]	F32
DDR_DQ[1]	F33
DDR_DQ[2]	J32
DDR_DQ[3]	J31
DDR_DQ[4]	E31
DDR_DQ[5]	E32
DDR_DQ[6]	H31

Signal	Ball
DDR_DQ[7]	H30
DDR_DQ[8]	K31
DDR_DQ[9]	K32
DDR_DQ[10]	M31
DDR_DQ[11]	N33
DDR_DQ[12]	H33
DDR_DQ[13]	J33
DDR_DQ[14]	M33
DDR_DQ[15]	M32
DDR_DQ[16]	N28
DDR_DQ[17]	N27
DDR_DQ[18]	T28
DDR_DQ[19]	U28
DDR_DQ[20]	M28
DDR_DQ[21]	M27
DDR_DQ[22]	T30
DDR_DQ[23]	T29
DDR_DQ[24]	P33
DDR_DQ[25]	P32
DDR_DQ[26]	U33
DDR_DQ[27]	U27
DDR_DQ[28]	N32
DDR_DQ[29]	N31
DDR_DQ[30]	T31
DDR_DQ[31]	T27
DDR_DQ[32]	W27
DDR_DQ[33]	W28
DDR_DQ[34]	AB26
DDR_DQ[35]	AB28
DDR_DQ[36]	Y25
DDR_DQ[37]	W26
DDR_DQ[38]	AA27
DDR_DQ[39]	AA28
DDR_DQ[40]	AD32
DDR_DQ[41]	AD33
DDR_DQ[42]	AF32
DDR_DQ[43]	AG32
DDR_DQ[44]	AC33
DDR_DQ[45]	AC31
DDR_DQ[46]	AF31
DDR_DQ[47]	AF33

Signal	Ball
DDR_DQ[48]	AC27
DDR_DQ[49]	AC26
DDR_DQ[50]	AE26
DDR_DQ[51]	AF26
DDR_DQ[52]	AB27
DDR_DQ[53]	AC28
DDR_DQ[54]	AF28
DDR_DQ[55]	AD26
DDR_DQ[56]	AH32
DDR_DQ[57]	AH33
DDR_DQ[58]	AL30
DDR_DQ[59]	AM30
DDR_DQ[60]	AG33
DDR_DQ[61]	AG31
DDR_DQ[62]	AL31
DDR_DQ[63]	AL32
DDR_DQS[0]	G32
DDR_DQS[0]#	G33
DDR_DQS[1]	L31
DDR_DQS[1]#	L32
DDR_DQS[2]	R29
DDR_DQS[2]#	R30
DDR_DQS[3]	T33
DDR_DQS[3]#	T32
DDR_DQS[4]	Y27
DDR_DQS[4]#	Y28
DDR_DQS[5]	AE32
DDR_DQS[5]#	AE33
DDR_DQS[6]	AE27
DDR_DQS[6]#	AE28
DDR_DQS[7]	AJ31
DDR_DQS[7]#	AH31
DDR_DQS[8]	AA32
DDR_DQS[8]#	AA33
DDR_DRVCRES	V33
DDR_ECC[0]	Y32
DDR_ECC[1]	Y33
DDR_ECC[2]	AB31
DDR_ECC[3]	AC32
DDR_ECC[4]	V32
DDR_ECC[5]	W31

Signal	Ball
DDR_ECC[6]	AB32
DDR_ECC[7]	AB33
DDR_ODT0	AL29
DDR_ODT1	L26
DDR_RAS#	AF27
DDR_RCOMPX	V26
DDR_SLEWCRES	V28
DDR_WE#	AN28
EECS	AH16
EEDI	AG16
EEDO	AK16
EESK	AF16
EX_ADDR[0]	AK9
EX_ADDR[1]	AH9
EX_ADDR[2]	AK8
EX_ADDR[3]	AJ9
EX_ADDR[4]	AM7
EX_ADDR[5]	AF10
EX_ADDR[6]	AN6
EX_ADDR[7]	AG9
EX_ADDR[8]	AN5
EX_ADDR[9]	AK5
EX_ADDR[10]	AM5
EX_ADDR[11]	AF9
EX_ADDR[12]	AK7
EX_ADDR[13]	AG8
EX_ADDR[14]	AL6
EX_ADDR[15]	AJ6
EX_ADDR[16]	AN4
EX_ADDR[17]	AG7
EX_ADDR[18]	AM4
EX_ADDR[19]	AF8
EX_ADDR[20]	AH7
EX_ADDR[21]	AH6
EX_ADDR[22]	AM3
EX_ADDR[23]	AJ5
EX_ADDR[24]	AL3
EX_ALE	AK4
EX_BE[0]#	AN7
EX_BE[1]#	AL7
EX_BURST	AN14

Signal	Ball
EX_CLK	AH10
EX_CS[0]#	AL10
EX_CS[1]#	AN9
EX_CS[2]#	AM9
EX_CS[3]#	AG10
EX_CS[4]#	AN8
EX_CS[5]#	AF11
EX_CS[6]#	AM8
EX_CS[7]#	AJ10
EX_DATA[0]	AN12
EX_DATA[1]	AF14
EX_DATA[2]	AL13
EX_DATA[3]	AH13
EX_DATA[4]	AK13
EX_DATA[5]	AG13
EX_DATA[6]	AM12
EX_DATA[7]	AJ13
EX_DATA[8]	AN11
EX_DATA[9]	AL12
EX_DATA[10]	AM11
EX_DATA[11]	AH12
EX_DATA[12]	AL11
EX_DATA[13]	AG12
EX_DATA[14]	AK11
EX_DATA[15]	AN10
NC57	AG14
EX_IOWAIT#	AN13
EX_PARITY[0]	AM14
EX_PARITY[1]	AL14
EX_RCOMPN	AG11
EX_RCOMPP	AJ11
EX_RD#	AK14
EX_RDY[0]#	AM15
EX_RDY[1]#	AF15
EX_RDY[2]#	AJ15
EX_RDY[3]#	AG15
Reserved19	AE15
Reserved20	AJ14
NC58	AH15
NC59	AD9
EX_WR#	AN15

Signal	Ball
GBE0_RxCLK	AL18
GBE0_RxCTL	AM18
GBE0_RXDATA[0]	AM20
GBE0_RXDATA[1]	AN20
GBE0_RXDATA[2]	AN19
GBE0_RXDATA[3]	AN18
GBE0_TxCLK	AL19
GBE0_TxCTL	AL20
GBE0_TXDATA[0]	AH18
GBE0_TXDATA[1]	AK20
GBE0_TXDATA[2]	AK19
GBE0_TXDATA[3]	AJ18
GBE1_RxCLK	AN21
GBE1_RxCTL	AM21
GBE1_RXDATA[0]	AK22
GBE1_RXDATA[1]	AN23
GBE1_RXDATA[2]	AN22
GBE1_RXDATA[3]	AL21
GBE1_TxCLK	AH19
GBE1_TxCTL	AF19
GBE1_TXDATA[0]	AK23
GBE1_TXDATA[1]	AF20
GBE1_TXDATA[2]	AK21
GBE1_TXDATA[3]	AH20
GBE2_RxCLK	AL22
GBE2_RxCTL	AN24
GBE2_RXDATA[0]	AL24
GBE2_RXDATA[1]	AH21
GBE2_RXDATA[2]	AL23
GBE2_RXDATA[3]	AM24
GBE2_TxCLK	AJ22
GBE2_TxCTL	AG21
GBE2_TXDATA[0]	AF22
GBE2_TXDATA[1]	AG22
GBE2_TXDATA[2]	AH22
GBE2_TXDATA[3]	AF21
GBE_AUX_PWR_GOOD	AN16
GBE_PME_WAKE	AE16
GBE_RCOMPN	AE17
GBE_RCOMPP	AF17
GBE_REFCLK	AN17

Signal	Ball
GBE_REFCLK_RMII	AL17
GP2_PIRQE#	D1
GP3_PIRQF#	H1
GP4_PIRQG#	N7
GP5_PIRQH#	G1
GP11_SMBALERT#	E4
GP16_IRQ24	H3
GP17_IRQ25	K3
GP18_IRQ36	K2
GP19_IRQ37	P8
GP20_IRQ26	P7
GP21_IRQ27	N4
GP23_IRQ28	L1
GP24_IRQ29	K7
GP25_IRQ38	J7
GP26_SATA0GP	P1
GP27_IRQ39	K8
GP28_IRQ30	J8
GP29_SATA1GP	G4
GP30_IRQ31	N5
GP31_IRQ32	W2
GP33_IRQ33	P4
GP34_IRQ34	T7
GP40_IRQ35	P3
GP41_LDRQ[1]#	P6
GPIO[0]	L6
GPIO[1]	J5
GPIO[6]	N9
GPIO[7]	J4
GPIO[8]	D5
GPIO[9]	D4
GPIO[10]	B4
GPIO[12]	H2
GPIO[13]	N8
GPIO[14]	G5
GPIO[15]	A4
GPIO[48]	U5
IERR#	AC4
INIT33V_OUT#	V1
INTRUDER#	L8
LAD[0]	N6

Signal	Ball
LAD[1]	L5
LAD[2]	L3
LAD[3]	L4
LDRQ[0]#	L2
LFRAME#	K1
MDC	AG17
MDIO	AH17
NC7	T8
NC9	W8
NC_SUS_TWO	J6
NC_TWO	U3
NC10	V9
NC11	W9
NC12	G19
NC13	H18
NC14	H19
NC15	G18
NC16	H20
NC17	E28
NC18	F28
NC19	D28
NC20	F12
NC21	B8
NC22	A8
NC34	E29
NC35	C31
NC36	K26
NC37	K27
NC38	D30
NC40	G31
NC41	L33
NC42	P27
NC43	R33
NC44	AA26
NC45	AE31
NC46	AD28
NC47	AJ33
NC48	K25
NC50	E30
NC51	J25
NC52	L27

Signal	Ball
NC53	AA31
NC54	D29
NC55	AL26
NC56	AE6
NMI	R5
OC[0]#	E12
OC[1]#	D12
PCICLK	N2
PCIRST#	G7
PE_HPINTR#	U2
PEA0_Rn[0]	B27
PEA0_Rn[1]	A26
PEA0_Rn[2]	B25
PEA0_Rn[3]	A24
PEA0_Rn[4]	A19
PEA0_Rn[5]	B20
PEA0_Rn[6]	A17
PEA0_Rn[7]	B18
PEA0_Rp[0]	A27
PEA0_Rp[1]	B26
PEA0_Rp[2]	A25
PEA0_Rp[3]	B24
PEA0_Rp[4]	B19
PEA0_Rp[5]	A20
PEA0_Rp[6]	B17
PEA0_Rp[7]	A18
PEA0_Tn[0]	E27
PEA0_Tn[1]	D26
PEA0_Tn[2]	E25
PEA0_Tn[3]	D24
PEA0_Tn[4]	D19
PEA0_Tn[5]	E20
PEA0_Tn[6]	D17
PEA0_Tn[7]	E18
PEA0_Tp[0]	D27
PEA0_Tp[1]	E26
PEA0_Tp[2]	D25
PEA0_Tp[3]	E24
PEA0_Tp[4]	E19
PEA0_Tp[5]	D20
PEA0_Tp[6]	E17

Signal	Ball
PEA0_Tp[7]	D18
PEA_CLKn	D22
PEA_CLKp	E22
PEA_ICOMPI	B22
PEA_ICOMPO	A22
PEA_RCOMPO	E21
PLTRST#	A5
PME#	B3
PROCHOT#	T6
PWRBTN#	H8
PWRGD	W6
PWROK	K6
RCIN#	M1
Reserved0	AB3
Reserved1	AC5
Reserved2	AH1
Reserved3	AB2
Reserved4	AB6
Reserved5	AB1
Reserved6	Y8
Reserved7	AA7
Reserved8	AA5
Reserved9	G21
Reserved10	D6
Reserved11	F3
Reserved12	AJ16
Reserved13	AL16
Reserved14	AM16
Reserved15	W7
Reserved16	Y6
Reserved17	B29
Reserved18	AB9
RI#	D8
RSMRST#	G3
RSTIN#	AC3
RTCX1	F1
RTCX2	F2
RTEST#	L7
Rx_CLK0	AC6
Rx_CLK1	AD2
Rx_CLK2	AE5

Signal	Ball
Rx_DATA_IN0	AE4
Rx_DATA_IN1	AE1
Rx_DATA_IN2	AD7
Rx_FRAME0	AC2
Rx_FRAME1	AB8
Rx_FRAME2	AE3
SATA_CLKREFn	G14
SATA_CLKREFp	G13
SATA_RBIAS	F15
SATA_RBIAS#	F16
SATA_RXn0	B14
SATA_RXn1	B15
SATA_RXp0	A14
SATA_RXp1	A15
SATA_TXn0	E14
SATA_TXn1	E15
SATA_TXp0	D14
SATA_TXp1	D15
SATALED#	J3
SERIRQ	J1
SIU_CTS1#	R1
SIU_CTS2#	T2
SIU_DCD1#	T5
SIU_DCD2#	T1
SIU_DSR1#	V5
SIU_DSR2#	W4
SIU_DTR1#	U7
SIU_DTR2#	AA2
SIU_RI1#	W3
SIU_RI2#	W1
SIU_RTS1#	U6
SIU_RTS2#	Y4
SIU_RXD1	R3
SIU_RXD2	T4
SIU_TXD1	R2
SIU_TXD2	T3
SLP_S3#	A6
SLP_S4#	C7
SLP_S5#	H9
SMBCLK	C5
SMBDATA	G6

Signal	Ball
SMBSCL	Y5
SMBSDA	Y1
SMI_OUT#	M2
SMLINK[0]	H7
SMLINK[1]	B5
SPI_CS#	AF2
SPI_MISO	AF1
SPI_MOSI	AC8
SPI_SCLK	AF3
SPKR	P2
SSP_EXTCLK	AH3
SSP_RXD	AF7
SSP_SCLK	AJ4
SSP_SFRM	AH4
SSP_TXD	AK3
STPCLK_OUT#	N3
SUS_STAT#	B6
SUSCLK	E7
SYS_PWR_OK	AJ17
SYS_RESET#	D7
TCK	Y3
TDI	AA1
TDO	AG2
THERMDA	F26
THERMDC	F27
THRMTRIP#	U4
TMS	AB5
TRST#	Y7
Tx_CLK0	AB4
Tx_CLK1	AF4
Tx_CLK2	AD5
Tx_DATA_OUT0	AC1
Tx_DATA_OUT1	AA9
Tx_DATA_OUT2	AE2
Tx_FRAME0	AB7
Tx_FRAME1	AD1
Tx_FRAME2	AD6
UART_CLK	U8
USB_RBIASn	A9
USB_RBIASp	B9
USBn0	A11

Signal	Ball
USBn1	A12
USBp0	B11
USBp1	B12
V_SEL	G2
VCC	K11
VCC	K13
VCC	L10
VCC	L12
VCC	L14
VCC	M11
VCC	M13
VCC	M15
VCC	N10
VCC	N12
VCC	N14
VCC	N16
VCC	P11
VCC	P13
VCC	P15
VCC	P17
VCC	R8
VCC	R10
VCC	R12
VCC	R14
VCC	R16
VCC	T11
VCC	T13
VCC	T15
VCC	T17
VCC	U10
VCC	U12
VCC	U14
VCC	U16
VCC	U18
VCC	U20
VCC	U22
VCC	U24
VCC	V11
VCC	V13
VCC	V15
VCC	V17

Signal	Ball
VCC	V19
VCC	V21
VCC	V23
VCC	W10
VCC	W12
VCC	W14
VCC	W16
VCC	W18
VCC	W20
VCC	W22
VCC	W24
VCC	Y11
VCC	Y13
VCC	Y15
VCC	Y17
VCC	Y19
VCC	Y21
VCC	Y23
VCC	AA10
VCC	AA12
VCC	AA14
VCC	AA16
VCC	AA18
VCC	AA20
VCC	AA22
VCC	AA24
VCC	AB11
VCC	AB13
VCC	AB15
VCC	AB17
VCC	AB19
VCC	AB21
VCC	AB23
VCC	AC10
VCC	AC12
VCC	AC14
VCC	AC16
VCC	AC18
VCC	AC20
VCC	AC22
VCC	AC24

Signal	Ball
VCC	AD11
VCC	AD13
VCC	AD15
VCC	AD17
VCC	AD19
VCC	AD21
VCC	AD23
VCC18	G30
VCC18	J29
VCC18	K30
VCC18	M30
VCC18	N29
VCC18	P30
VCC18	R27
VCC18	T25
VCC18	W30
VCC18	Y29
VCC18	AA30
VCC18	AC30
VCC18	AD29
VCC18	AE30
VCC18	AG30
VCC18	AH29
VCC18	AJ30
VCC18	AK25
VCC18	AM25
VCC18	AM31
VCC1P2_USBSUS	J12
VCC25	AE20
VCC25	AJ19
VCC25	AJ21
VCC25	AM22
VCC33	E5
VCC33	H4
VCC33	H6
VCC33	M4
VCC33	R6
VCC33	R9
VCC33	V4
VCC33	AA6
VCC33	AD4

Signal	Ball
VCC33	AE9
VCC33	AF6
VCC33	AH11
VCC33	AL5
VCC33	AL9
VCC33	AL15
VCC50	K9
VCC50	P9
VCC50	U9
VCC50	Y9
VCC50	AC9
VCC50	AE11
VCC50	AE14
VCC50_SUS	K4
VCCA[1]	G24
VCCA[2]	F24
VCCABG3P3_USB	H12
VCCABGP033	G15
VCCAHPLL	T19
VCCAPE	A28
VCCAPE	B21
VCCAPE	C19
VCCAPE	C22
VCCAPE	C24
VCCAPE	C28
VCCAPE	F19
VCCAPE	F21
VCCAPE	F23
VCCAPE	F25
VCCAPE	G16
VCCAPE	G20
VCCAPE0PLL12	F18
VCCAPE0PLL12	F20
VCCAPLL	C15
VCCARX	B13
VCCARX	C14
VCCARX	F14
VCCASATABG3P3	H15
VCCATX	B16
VCCATX	C16
VCCATX	E16

Signal	Ball
VCCAUSB12	D13
VCCAUSB12	F13
VCCAUSB12	J13
VCCGBE33	AE19
VCCGBEPSUS	AG20
VCCPRTC	M5
VCCPSUS	F10
VCCPSUS	G11
VCCPSUS	J11
VCCRPE	B23
VCCRPE	C18
VCCRPE	C26
VCCRPE	D21
VCCRPE	D23
VCCRPE	E23
VCCRPE	F17
VCCRPE	F22
VCCSATA	H14
VCCSATA	J14
VCCSATA33	C13
VCCSUS1	H10
VCCSUS1	J10
VCCSUS1	M9
VCCSUS1	M10
VCCSUS25	AE18
VCCSUS25	AG18
VCCSUS25	AK18
VCCSUS25	AM17
VCCTMP18	N25
VCCUSB12	B7
VCCUSB12	B10
VCCUSB12	C8
VCCUSB12	C11
VCCUSB12	E9
VCCUSB12	F11
VCCUSB12	G10
VCCVC	A29
VCCVC	B30
VCCVC	G22
VCCVC	H21
VCCVC	H23

Signal	Ball
VCCVC	H24
VCCVC	H26
VCCVC	H27
VCCVC	J16
VCCVC	J18
VCCVC	J20
VCCVC	J22
VCCVC	J24
VCCVC	J26
VCCVC	K15
VCCVC	K17
VCCVC	K19
VCCVC	K21
VCCVC	K23
VCCVC	L16
VCCVC	L18
VCCVC	L20
VCCVC	L22
VCCVC	L24
VCCVC	M17
VCCVC	M19
VCCVC	M21
VCCVC	M23
VCCVC	N18
VCCVC	N20
VCCVC	N22
VCCVC	N24
VCCVC	P19
VCCVC	P21
VCCVC	P23
VCCVC	P25
VCCVC	R18
VCCVC	R20
VCCVC	R22
VCCVC	R24
VCCVC	T21
VCCVC	T23
VCCVC	T24
VCCVC	U23
VCCVC	U25
VCCVC	U26

Signal	Ball
VRMPWRGD	V7
VSS	A2
VSS	A3
VSS	A7
VSS	A10
VSS	A13
VSS	A16
VSS	A21
VSS	A23
VSS	A30
VSS	A31
VSS	A32
VSS	A33
VSS	B1
VSS	B2
VSS	B28
VSS	B32
VSS	B33
VSS	C1
VSS	C2
VSS	C3
VSS	C4
VSS	C6
VSS	C9
VSS	C10
VSS	C12
VSS	C17
VSS	C20
VSS	C21
VSS	C23
VSS	C25
VSS	C27
VSS	C29
VSS	C33
VSS	D2
VSS	D3
VSS	D9
VSS	D10
VSS	D16
VSS	D33
VSS	E1

Signal	Ball
VSS	E2
VSS	E3
VSS	E6
VSS	E8
VSS	E10
VSS	E11
VSS	E13
VSS	F5
VSS	F6
VSS	F7
VSS	F8
VSS	F9
VSS	F30
VSS	G8
VSS	G9
VSS	G12
VSS	G17
VSS	G23
VSS	G25
VSS	G29
VSS	H5
VSS	H11
VSS	H13
VSS	H16
VSS	H17
VSS	H22
VSS	H25
VSS	H29
VSS	H32
VSS	J2
VSS	J9
VSS	J15
VSS	J17
VSS	J19
VSS	J21
VSS	J23
VSS	J30
VSS	K5
VSS	K10
VSS	K12
VSS	K14

Signal	Ball
VSS	K16
VSS	K18
VSS	K20
VSS	K22
VSS	K24
VSS	K29
VSS	L9
VSS	L11
VSS	L13
VSS	L15
VSS	L17
VSS	L19
VSS	L21
VSS	L23
VSS	L25
VSS	L28
VSS	L30
VSS	M3
VSS	M6
VSS	M12
VSS	M14
VSS	M16
VSS	M18
VSS	M20
VSS	M22
VSS	M24
VSS	M29
VSS	N11
VSS	N13
VSS	N15
VSS	N17
VSS	N19
VSS	N21
VSS	N23
VSS	N26
VSS	N30
VSS	P10
VSS	P12
VSS	P14
VSS	P16
VSS	P18

Signal	Ball
VSS	P20
VSS	P22
VSS	P24
VSS	P26
VSS	P29
VSS	P31
VSS	R4
VSS	R7
VSS	R11
VSS	R13
VSS	R15
VSS	R17
VSS	R19
VSS	R21
VSS	R23
VSS	R25
VSS	R26
VSS	R28
VSS	R31
VSS	T9
VSS	T10
VSS	T12
VSS	T14
VSS	T16
VSS	T18
VSS	T20
VSS	T22
VSS	T26
VSS	U11
VSS	U13
VSS	U15
VSS	U17
VSS	U21
VSS	V3
VSS	V6
VSS	V8
VSS	V10
VSS	V12
VSS	V14
VSS	V16
VSS	V18

Signal	Ball
VSS	V20
VSS	V25
VSS	W11
VSS	W13
VSS	W15
VSS	W17
VSS	W19
VSS	W21
VSS	W23
VSS	W25
VSS	W29
VSS	Y10
VSS	Y12
VSS	Y14
VSS	Y16
VSS	Y18
VSS	Y20
VSS	Y22
VSS	Y24
VSS	Y30
VSS	AA4
VSS	AA8
VSS	AA11
VSS	AA13
VSS	AA15
VSS	AA17
VSS	AA19
VSS	AA21
VSS	AA23
VSS	AA29
VSS	AB10
VSS	AB12
VSS	AB14
VSS	AB16
VSS	AB18
VSS	AB20
VSS	AB22
VSS	AB24
VSS	AB25
VSS	AB30
VSS	AC7

Signal	Ball
VSS	AC11
VSS	AC13
VSS	AC15
VSS	AC17
VSS	AC19
VSS	AC21
VSS	AC23
VSS	AC29
VSS	AD3
VSS	AD10
VSS	AD12
VSS	AD14
VSS	AD16
VSS	AD18
VSS	AD20
VSS	AD22
VSS	AD24
VSS	AD25
VSS	AD30
VSS	AE8
VSS	AE10
VSS	AE12
VSS	AE13
VSS	AE21
VSS	AE22
VSS	AE23
VSS	AE25
VSS	AE29
VSS	AF12
VSS	AF13
VSS	AF18
VSS	AF24
VSS	AF25
VSS	AF30
VSS	AG4
VSS	AG6
VSS	AG19
VSS	AG23
VSS	AG24
VSS	AG28
VSS	AG29

Signal	Ball
VSS	AH5
VSS	AH8
VSS	AH14
VSS	AH23
VSS	AH24
VSS	AH25
VSS	AH30
VSS	AJ2
VSS	AJ7
VSS	AJ8
VSS	AJ12
VSS	AJ20
VSS	AJ23
VSS	AJ24
VSS	AJ29
VSS	AK6
VSS	AK10
VSS	AK12
VSS	AK15
VSS	AK17
VSS	AK24
VSS	AK30
VSS	AK31
VSS	AK32
VSS	AK33
VSS	AL1
VSS	AL4
VSS	AL8
VSS	AL25
VSS	AL33
VSS	AM1
VSS	AM2
VSS	AM6
VSS	AM10
VSS	AM13
VSS	AM19
VSS	AM23
VSS	AM32
VSS	AM33
VSS	AN1
VSS	AN2

Signal	Ball
VSS	AN3
VSS	AN25
VSS	AN30
VSS	AN31
VSS	AN32
VSS	AN33
VSSA	U19
VTTDDR	F29
VTTDDR	L29
VTTDDR	V22
VTTDDR	V24
VTTDDR	V29
VTTDDR	AA25
VTTDDR	AB29
VTTDDR	AC25
VTTDDR	AE24
VTTDDR	AF23
VTTDDR	AF29
VTTDDR	AG25
VTTDDR	AJ25
VTTDDR	AK29
WDT_TOUT#	F4

Table 48-34.EP80579 Ball Map (bottom view, left side) (Sheet 1 of 2)

	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T
1		VSS	VSS	GP2_PIRQE#	VSS	RTCX1	GP5_PIRQH#	GP3_PIRQF#	SERIRQ	LFRAME#	GP23_IRQ28	RCIN#	CLK14	GP26_SATA0GP	SIU_CTS1#	SIU_DCD2#
2	VSS	VSS	VSS	VSS	VSS	RTCX2	V_SEL	GPIO[12]	VSS	GP18_IRQ36	LDRQ[0]#	SMI_OUT#	PCICLK	SPKR	SIU_TXD1	SIU_CTS2#
3	VSS	PME#	VSS	VSS	VSS	Reserved11	RSMRST#	GP16_IRQ24	SATALED#	GP17_IRQ25	LAD[2]	VSS	STPCLK_OUT#	GP40_IRQ35	SIU_RXD1	SIU_TXD2
4	GPIO[15]	GPIO[10]	VSS	GPIO[9]	GP11_SMBALERT#	WDT_TOUT#	GP29_SATA1GP	VCC33	GPIO[7]	VCC50_SUS	LAD[3]	VCC33	GP21_IRQ27	GP33_IRQ33	VSS	SIU_RXD2
5	PLTRST#	SMLINK[1]	SMBCLK	GPIO[8]	VCC33	VSS	GPIO[14]	VSS	GPIO[1]	VSS	LAD[1]	VCCPRTC	GP30_IRQ31	CPUSLP_OUT#	NMI	SIU_DCD1#
6	SLP_S3#	SUS_STAT#	VSS	Reserved10	VSS	VSS	SMBDATA	VCC33	NC_SUS_TWO	PWROK	GPIO[0]	VSS	LAD[0]	GP41_LDRQ[1]#	VCC33	PROCHOT#
7	VSS	VCCUSB12	SLP_S4#	SYS_RESET#	SUSCLK	VSS	PCIRST#	SMLINK[0]	GP25_IRQ38	GP24_IRQ29	RTEST#	CPUPWRGD_OUT	GP4_PIRQG#	GP20_IRQ26	VSS	GP34_IRQ34
8	NC22	NC21	VCCUSB12	RI#	VSS	VSS	VSS	PWRBTN#	GP28_IRQ30	GP27_IRQ39	INTRUDER#	BSEL	GPIO[13]	GP19_IRQ37	VCC	NC7
9	USB_RBIASn	USB_RBIASp	VSS	VSS	VCCUSB12	VSS	VSS	SLP_S5#	VSS	VCC50	VSS	VCCSUS1	GPIO[6]	VCC50	VCC33	VSS
10	VSS	VCCUSB12	VSS	VSS	VSS	VCCPSUS	VCCUSB12	VCCSUS1	VCCSUS1	VSS	VCC	VCCSUS1	VCC	VSS	VCC	VSS
11	USBn0	USBp0	VCCUSB12	CLK48	VSS	VCCUSB12	VCCPSUS	VSS	VCCPSUS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
12	USBn1	USBp1	VSS	OC[1]#	OC[0]#	NC20	VSS	VCCABG3P3_USB	VCC1P2_USBSUS	VSS	VCC	VSS	VCC	VSS	VCC	VSS
13	VSS	VCCARX	VCCSATA33	VCCAUSB12	VSS	VCCAUSB12	SATA_CLKREFp	VSS	VCCAUSB12	VCC	VSS	VCC	VSS	VCC	VSS	VCC
14	SATA_RXp0	SATA_RXn0	VCCARX	SATA_TXp0	SATA_TXn0	VCCARX	SATA_CLKREFn	VCCSATA	VCCSATA	VSS	VCC	VSS	VCC	VSS	VCC	VSS
15	SATA_RXp1	SATA_RXn1	VCCAPLL	SATA_TXp1	SATA_TXn1	SATA_RBIAS	VCCABGP033	VCCASATABG3P3	VSS	VCCVC	VSS	VCC	VSS	VCC	VSS	VCC
16	VSS	VCCATX	VCCATX	VSS	VCCATX	SATA_RBIAS#	VCCAPE	VSS	VCCVC	VSS	VCCVC	VSS	VCC	VSS	VCC	VSS
17	PEA0_Rn[6]	PEA0_Rp[6]	VSS	PEA0_Tn[6]	PEA0_Tp[6]	VCCRPE	VSS	VSS	VSS	VCCVC	VSS	VCCVC	VSS	VCC	VSS	VCC
18	PEA0_Rp[7]	PEA0_Rn[7]	VCCRPE	PEA0_Tp[7]	PEA0_Tn[7]	VCCAPE0PLL12	NC15	NC13	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS
19	PEA0_Rn[4]	PEA0_Rp[4]	VCCAPE	PEA0_Tn[4]	PEA0_Tp[4]	VCCAPE	NC12	NC14	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCAHPLL
20	PEA0_Rp[5]	PEA0_Rn[5]	VSS	PEA0_Tp[5]	PEA0_Tn[5]	VCCAPE0PLL12	VCCAPE	NC16	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS

Table 48-34.EP80579 Ball Map (bottom view, left side) (Sheet 2 of 2)

	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T
21	VSS	VCCAPE	VSS	VCCRPE	PEA_RCOMPO	VCCAPE	Reserved9	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC
22	PEA_ICOMPO	PEA_ICOMPI	VCCAPE	PEA_CLKn	PEA_CLKp	VCCRPE	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS
23	VSS	VCCRPE	VSS	VCCRPE	VCCRPE	VCCAPE	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC
24	PEA0_Rn[3]	PEA0_Rp[3]	VCCAPE	PEA0_Tn[3]	PEA0_Tp[3]	VCCA[2]	VCCA[1]	VCCVC	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VCCVC
25	PEA0_Rp[2]	PEA0_Rn[2]	VSS	PEA0_Tp[2]	PEA0_Tn[2]	VCCAPE	VSS	VSS	NC51	NC48	VSS	DDR_CKE[1]	VCCTMP18	VCCVC	VSS	VCC18
26	PEA0_Rn[1]	PEA0_Rp[1]	VCCRPE	PEA0_Tn[1]	PEA0_Tp[1]	THERMDA	DDR_CK[5]	VCCVC	VCCVC	NC36	DDR_ODT1	BPM1	VSS	VSS	VSS	VSS
27	PEA0_Rp[0]	PEA0_Rn[0]	VSS	PEA0_Tp[0]	PEA0_Tn[0]	THERMDC	DDR_CK[5]#	VCCVC	DDR_CK[3]#	NC37	NC52	DDR_DQ[21]	DDR_DQ[17]	NC42	VCC18	DDR_DQ[31]
28	VCCAPE	VSS	VCCAPE	NC19	NC17	NC18	BPM3	DDR_CS[0]#	DDR_CK[3]	DDR_CS[1]#	VSS	DDR_DQ[20]	DDR_DQ[16]	DDR_DM[2]	VSS	DDR_DQ[18]
29	VCCVC	Reserved17	VSS	NC54	NC34	VTTDDR	VSS	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	DDR_DQS[2]	DDR_DQ[23]
30	VSS	VCCVC	CLKP100	NC38	NC50	VSS	VCC18	DDR_DQ[7]	VSS	VCC18	VSS	VCC18	VSS	VCC18	DDR_DQS[2]#	DDR_DQ[22]
31	VSS	CLKN100	NC35	DDR_CK[4]	DDR_DQ[4]	DDR_DM[0]	NC40	DDR_DQ[6]	DDR_DQ[3]	DDR_DQ[8]	DDR_DQS[1]	DDR_DQ[10]	DDR_DQ[29]	VSS	VSS	DDR_DQ[30]
32	VSS	VSS	BPM2	DDR_CK[4]#	DDR_DQ[5]	DDR_DQ[0]	DDR_DQS[0]	VSS	DDR_DQ[2]	DDR_DQ[9]	DDR_DQS[1]#	DDR_DQ[15]	DDR_DQ[28]	DDR_DQ[25]	DDR_DM[3]	DDR_DQS[3]#
33	VSS	VSS	VSS	VSS	BPM0	DDR_DQ[1]	DDR_DQS[0]#	DDR_DQ[12]	DDR_DQ[13]	DDR_DM[1]	NC41	DDR_DQ[14]	DDR_DQ[11]	DDR_DQ[24]	NC43	DDR_DQS[3]

Table 48-35. EP80579 Ball Map (bottom view, right side) (Sheet 1 of 2)

U	V	W	Y	AA	AB	AC	AD	AE	AF	AG	AH	AJ	AK	AL	AM	AN	
CPURST#	INIT33V_OUT#	SIU_RI2#	SMBSDA	TDI	Reserved5	Tx_DATA_OUT0	Tx_FRAME1	Rx_DATA_IN1	SPI_MISO	1588_RX_SNAP	Reserved2	AMMSSIG	1588_PPS	VSS	VSS	VSS	1
PE_HPINTR#	A20GATE	GP31_IRQ32	BPM3_IN	SIU_DTR2#	Reserved3	Rx_FRAME0	Rx_CLK1	Tx_DATA_OUT2	SPI_CS#	TDO	ASMSSIG	VSS	CN1RXD	1588_TESTMODE_DATA	VSS	VSS	2
NC_TWO	VSS	SIU_RI1#	TCK	BPM4_PRDY_OUT	Reserved0	RSTIN#	VSS	Rx_FRAME2	SPI_SCLK	1588_TX_SNAP	SSP_EXTCLK	CN1TXD	SSP_TXD	EX_ADDR[24]	EX_ADDR[22]	VSS	3
THRMTRIP#	VCC33	SIU_DSR2#	SIU_RTS2#	VSS	Tx_CLK0	IERR#	VCC33	Rx_DATA_IN0	Tx_CLK1	VSS	SSP_SFRM	SSP_SCLK	EX_ALE	VSS	EX_ADDR[18]	EX_ADDR[16]	4
GPIO[48]	SIU_DSR1#	BPM5_PREQ_IN	SMBSCL	Reserved8	TMS	Reserved1	Tx_CLK2	Rx_CLK2	CN0RXD	CN1TXEN	VSS	EX_ADDR[23]	EX_ADDR[9]	VCC33	EX_ADDR[10]	EX_ADDR[8]	5
SIU_RTS1#	VSS	PWRGD	Reserved16	VCC33	Reserved4	Rx_CLK0	Tx_FRAME2	NC56	VCC33	VSS	EX_ADDR[21]	EX_ADDR[15]	VSS	EX_ADDR[14]	VSS	EX_ADDR[6]	6
SIU_DTR1#	VRMPWRGD	Reserved15	TRST#	Reserved7	Tx_FRAME0	VSS	Rx_DATA_IN2	CN0TXEN	SSP_RXD	EX_ADDR[17]	EX_ADDR[20]	VSS	EX_ADDR[12]	EX_BE[1]#	EX_ADDR[4]	EX_BE[0]#	7
UART_CLK	VSS	NC9	Reserved6	VSS	Rx_FRAME1	SPI_MOSI	CN0TXD	VSS	EX_ADDR[19]	EX_ADDR[13]	VSS	VSS	EX_ADDR[2]	VSS	EX_CS[6]#	EX_CS[4]#	8
VCC50	NC10	NC11	VCC50	Tx_DATA_OUT1	Reserved18	VCC50	NC59	VCC33	EX_ADDR[11]	EX_ADDR[7]	EX_ADDR[1]	EX_ADDR[3]	EX_ADDR[0]	VCC33	EX_CS[2]#	EX_CS[1]#	9
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VSS	EX_ADDR[5]	EX_CS[3]#	EX_CLK	EX_CS[7]#	VSS	EX_CS[0]#	VSS	EX_DATA[15]	10
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCC50	EX_CS[5]#	EX_RCOMPN	VCC33	EX_RCOMPP	EX_DATA[14]	EX_DATA[12]	EX_DATA[10]	EX_DATA[8]	11
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VSS	VSS	EX_DATA[13]	EX_DATA[11]	VSS	VSS	EX_DATA[9]	EX_DATA[6]	EX_DATA[0]	12
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VSS	EX_DATA[5]	EX_DATA[3]	EX_DATA[7]	EX_DATA[4]	EX_DATA[2]	VSS	EX_IOWAIT#	13
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC50	EX_DATA[1]	NC57	VSS	Reserved20	EX_RD#	EX_PARITY[1]	EX_PARITY[0]	EX_BURST	14
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	Reserved19	EX_RDY[1]#	EX_RDY[3]#	NC58	EX_RDY[2]#	VSS	VCC33	EX_RDY[0]#	EX_WR#	15
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	GBE_PME_WAKE	EESK	EEDI	EECS	Reserved12	EEDO	Reserved13	Reserved14	GBE_AUX_PWR_GOOD	16
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	GBE_RCOMPN	GBE_RCOMPP	MDC	MDIO	SYS_PWR_OK	VSS	GBE_REFCLK_RMII	VCCSUS25	GBE_REFCLK	17

Table 48-35.EP80579 Ball Map (bottom view, right side) (Sheet 2 of 2)

U	V	W	Y	AA	AB	AC	AD	AE	AF	AG	AH	AJ	AK	AL	AM	AN	
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCSUS25	VSS	VCCSUS25	GBE0_TXDATA[0]	GBE0_TXDATA[3]	VCCSUS25	GBE0_RxCLK	GBE0_RxCTL	GBE0_RXDATA[3]	18
VSSA	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCGBE33	GBE1_TxCTL	VSS	GBE1_TxCLK	VCC25	GBE0_TXDATA[2]	GBE0_TxCLK	VSS	GBE0_RXDATA[2]	19
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC25	GBE1_TXDATA[1]	VCCGBEPSUS	GBE1_TXDATA[3]	VSS	GBE0_TXDATA[1]	GBE0_TxCTL	GBE0_RXDATA[0]	GBE0_RXDATA[1]	20
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	GBE2_TXDATA[3]	GBE2_TxCTL	GBE2_RXDATA[1]	VCC25	GBE1_TXDATA[2]	GBE1_RXDATA[3]	GBE1_RxCTL	GBE1_RxCLK	21
VCC	VTTDDR	VCC	VSS	VCC	VSS	VCC	VSS	VSS	GBE2_TXDATA[0]	GBE2_TXDATA[1]	GBE2_TXDATA[2]	GBE2_TxCLK	GBE1_RXDATA[0]	GBE2_RxCLK	VCC25	GBE1_RXDATA[2]	22
VCCVC	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VTTDDR	VSS	VSS	VSS	GBE1_TXDATA[0]	GBE2_RXDATA[2]	VSS	GBE1_RXDATA[1]	23
VCC	VTTDDR	VCC	VSS	VCC	VSS	VCC	VSS	VTTDDR	VSS	VSS	VSS	VSS	VSS	GBE2_RXDATA[0]	GBE2_RXDATA[3]	GBE2_RxCTL	24
VCCVC	VSS	VSS	DDR_DQ[36]	VTTDDR	VSS	VTTDDR	VSS	VSS	VSS	VTTDDR	VSS	VTTDDR	VCC18	VSS	VCC18	VSS	25
VCCVC	DDR_RCOMPX	DDR_DQ[37]	DDR_DM[4]	NC44	DDR_DQ[34]	DDR_DQ[49]	DDR_DQ[55]	DDR_DQ[50]	DDR_DQ[51]	DDR_BA[2]	DDR_A[12]	DDR_A[7]	DDR_A[5]	NC55	DDR_A[0]	DDR_CAS#	26
DDR_DQ[27]	DDR_CRES[0]	DDR_DQ[32]	DDR_DQS[4]	DDR_DQ[38]	DDR_DQ[52]	DDR_DQ[48]	DDR_DM[6]	DDR_DQS[6]	DDR_RAS#	DDR_A[14]	DDR_A[9]	DDR_A[8]	DDR_A[4]	DDR_A[1]	DDR_A[10]	DDR_A[13]	27
DDR_DQ[19]	DDR_SLEWCRES	DDR_DQ[33]	DDR_DQS[4]#	DDR_DQ[39]	DDR_DQ[35]	DDR_DQ[53]	NC46	DDR_DQS[6]#	DDR_DQ[54]	VSS	DDR_A[11]	DDR_A[6]	DDR_A[3]	DDR_A[2]	DDR_BA[0]	DDR_WE#	28
DDR_CRES[2]	VTTDDR	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	VTTDDR	DDR_ODT0	DDR_BA[1]	DDR_CKE[0]	29
DDR_CRES[1]	DDR_CK[0]	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	DDR_DQ[58]	DDR_DQ[59]	VSS	30
DDR_CK[2]#	DDR_CK[0]#	DDR_ECC[5]	DDR_DM[8]	NC53	DDR_ECC[2]	DDR_DQ[45]	DDR_DM[5]	NC45	DDR_DQ[46]	DDR_DQ[61]	DDR_DQS[7]#	DDR_DQS[7]	VSS	DDR_DQ[62]	VCC18	VSS	31
DDR_CK[2]	DDR_ECC[4]	DDR_CK[1]	DDR_ECC[0]	DDR_DQS[8]	DDR_ECC[6]	DDR_ECC[3]	DDR_DQ[40]	DDR_DQS[5]	DDR_DQ[42]	DDR_DQ[43]	DDR_DQ[56]	DDR_DM[7]	VSS	DDR_DQ[63]	VSS	VSS	32
DDR_DQ[26]	DDR_DRVCRES	DDR_CK[1]#	DDR_ECC[1]	DDR_DQS[8]#	DDR_ECC[7]	DDR_DQ[44]	DDR_DQ[41]	DDR_DQS[5]#	DDR_DQ[47]	DDR_DQ[60]	DDR_DQ[57]	NC47	VSS	VSS	VSS	VSS	33

Table 48-36.EP80579 Ball Map (top view, left side) (Sheet 1 of 2)

	AN	AM	AL	AK	AJ	AH	AG	AF	AE	AD	AC	AB	AA	Y	W	V	U
1	VSS	VSS	VSS	1588_PPS	AMMSSIG	Reserved2	1588_RX_SNAP	SPI_MISO	Rx_DATA_IN1	Tx_FRAME1	Tx_DATA_OUT0	Reserved5	TDI	SMBSDA	SIU_RI2#	INIT33V_OUT#	CPURST#
2	VSS	VSS	1588_TESTMODE_DATA	CN1RXD	VSS	ASMSSIG	TDO	SPI_CS#	Tx_DATA_OUT2	Rx_CLK1	Rx_FRAME0	Reserved3	SIU_DTR2#	BPM3_IN	GP31_IRQ32	A20GATE	PE_HPINTR#
3	VSS	EX_ADDR[22]	EX_ADDR[24]	SSP_TXD	CN1TXD	SSP_EXTCLK	1588_TX_SNAP	SPI_SCLK	Rx_FRAME2	VSS	RSTIN#	Reserved0	BPM4_PRDY_OUT	TCK	SIU_RI1#	VSS	NC_TWO
4	EX_ADDR[16]	EX_ADDR[18]	VSS	EX_ALE	SSP_SCLK	SSP_SFRM	VSS	Tx_CLK1	Rx_DATA_IN0	VCC33	IERR#	Tx_CLK0	VSS	SIU_RTS2#	SIU_DSR2#	VCC33	THRMTRIP#
5	EX_ADDR[8]	EX_ADDR[10]	VCC33	EX_ADDR[9]	EX_ADDR[23]	VSS	CN1TXEN	CN0RXD	Rx_CLK2	Tx_CLK2	Reserved1	TMS	Reserved8	SMBSCL	BPM5_PREQ_IN	SIU_DSR1#	GPIO[48]
6	EX_ADDR[6]	VSS	EX_ADDR[14]	VSS	EX_ADDR[15]	EX_ADDR[21]	VSS	VCC33	NC56	Tx_FRAME2	Rx_CLK0	Reserved4	VCC33	Reserved16	PWRGD	VSS	SIU_RTS1#
7	EX_BE[0]#	EX_ADDR[4]	EX_BE[1]#	EX_ADDR[12]	VSS	EX_ADDR[20]	EX_ADDR[17]	SSP_RXD	CN0TXEN	Rx_DATA_IN2	VSS	Tx_FRAME0	Reserved7	TRST#	Reserved15	VRMPWRGD	SIU_DTR1#
8	EX_CS[4]#	EX_CS[6]#	VSS	EX_ADDR[2]	VSS	VSS	EX_ADDR[13]	EX_ADDR[19]	VSS	CN0TXD	SPI_MOSI	Rx_FRAME1	VSS	Reserved6	NC9	VSS	UART_CLK
9	EX_CS[1]#	EX_CS[2]#	VCC33	EX_ADDR[0]	EX_ADDR[3]	EX_ADDR[1]	EX_ADDR[7]	EX_ADDR[11]	VCC33	NC59	VCC50	Reserved18	Tx_DATA_OUT1	VCC50	NC11	NC10	VCC50
10	EX_DATA[15]	VSS	EX_CS[0]#	VSS	EX_CS[7]#	EX_CLK	EX_CS[3]#	EX_ADDR[5]	VSS	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
11	EX_DATA[8]	EX_DATA[10]	EX_DATA[12]	EX_DATA[14]	EX_RCOMP	VCC33	EX_RCOMPN	EX_CS[5]#	VCC50	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS
12	EX_DATA[0]	EX_DATA[6]	EX_DATA[9]	VSS	VSS	EX_DATA[11]	EX_DATA[13]	VSS	VSS	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
13	EX_IOWAIT#	VSS	EX_DATA[2]	EX_DATA[4]	EX_DATA[7]	EX_DATA[3]	EX_DATA[5]	VSS	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS
14	EX_BURST	EX_PARITY[0]	EX_PARITY[1]	EX_RD#	Reserved20	VSS	NC57	EX_DATA[1]	VCC50	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
15	EX_WR#	EX_RDY[0]#	VCC33	VSS	EX_RDY[2]#	NC58	EX_RDY[3]#	EX_RDY[1]#	Reserved19	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS
16	GBE_AUX_PWR_GOOD	Reserved14	Reserved13	EEDO	Reserved12	EECS	EEDI	EESK	GBE_PME_WAKE	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
17	GBE_REFCLK	VCCSUS25	GBE_REFCLK_RMII	VSS	SYS_PWR_OK	MDIO	MDC	GBE_RCOMPP	GBE_RCOMPN	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS

Table 48-36.EP80579 Ball Map (top view, left side) (Sheet 2 of 2)

	AN	AM	AL	AK	AJ	AH	AG	AF	AE	AD	AC	AB	AA	Y	W	V	U
18	GBE0_RXDATA[3]	GBE0_RxCTL	GBE0_RxCLK	VCCSUS25	GBE0_TXDATA[3]	GBE0_TXDATA[0]	VCCSUS25	VSS	VCCSUS25	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
19	GBE0_RXDATA[2]	VSS	GBE0_TxCLK	GBE0_TXDATA[2]	VCC25	GBE1_TxCLK	VSS	GBE1_TxCTL	VCCGBE33	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSSA
20	GBE0_RXDATA[1]	GBE0_RXDATA[0]	GBE0_TxCTL	GBE0_TXDATA[1]	VSS	GBE1_TXDATA[3]	VCCGBEPSUS	GBE1_TXDATA[1]	VCC25	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC
21	GBE1_RxCLK	GBE1_RxCTL	GBE1_RXDATA[3]	GBE1_TXDATA[2]	VCC25	GBE2_RXDATA[1]	GBE2_TxCTL	GBE2_TXDATA[3]	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS
22	GBE1_RXDATA[2]	VCC25	GBE2_RxCLK	GBE1_RXDATA[0]	GBE2_TxCLK	GBE2_TXDATA[2]	GBE2_TXDATA[1]	GBE2_TXDATA[0]	VSS	VSS	VCC	VSS	VCC	VSS	VCC	VTTDDR	VCC
23	GBE1_RXDATA[1]	VSS	GBE2_RXDATA[2]	GBE1_TXDATA[0]	VSS	VSS	VSS	VTTDDR	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCVC
24	GBE2_RxCTL	GBE2_RXDATA[3]	GBE2_RXDATA[0]	VSS	VSS	VSS	VSS	VSS	VTTDDR	VSS	VCC	VSS	VCC	VSS	VCC	VTTDDR	VCC
25	VSS	VCC18	VSS	VCC18	VTTDDR	VSS	VTTDDR	VSS	VSS	VSS	VTTDDR	VSS	VTTDDR	DDR_DQ[36]	VSS	VSS	VCCVC
26	DDR_CAS#	DDR_A[0]	NC55	DDR_A[5]	DDR_A[7]	DDR_A[12]	DDR_BA[2]	DDR_DQ[51]	DDR_DQ[50]	DDR_DQ[55]	DDR_DQ[49]	DDR_DQ[34]	NC44	DDR_DM[4]	DDR_DQ[37]	DDR_RCOMPX	VCCVC
27	DDR_A[13]	DDR_A[10]	DDR_A[1]	DDR_A[4]	DDR_A[8]	DDR_A[9]	DDR_A[14]	DDR_RAS#	DDR_DQS[6]	DDR_DM[6]	DDR_DQ[48]	DDR_DQ[52]	DDR_DQ[38]	DDR_DQS[4]	DDR_DQ[32]	DDR_CRES[0]	DDR_DQ[27]
28	DDR_WE#	DDR_BA[0]	DDR_A[2]	DDR_A[3]	DDR_A[6]	DDR_A[11]	VSS	DDR_DQ[54]	DDR_DQS[6]#	NC46	DDR_DQ[53]	DDR_DQ[35]	DDR_DQ[39]	DDR_DQS[4]#	DDR_DQ[33]	DDR_SLEWCRES	DDR_DQ[19]
29	DDR_CKE[0]	DDR_BA[1]	DDR_ODT0	VTTDDR	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	VTTDDR	DDR_CRES[2]
30	VSS	DDR_DQ[59]	DDR_DQ[58]	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	VSS	VCC18	DDR_CK[0]	DDR_CRES[1]
31	VSS	VCC18	DDR_DQ[62]	VSS	DDR_DQS[7]	DDR_DQS[7]#	DDR_DQ[61]	DDR_DQ[46]	NC45	DDR_DM[5]	DDR_DQ[45]	DDR_ECC[2]	NC53	DDR_DM[8]	DDR_ECC[5]	DDR_CK[0]#	DDR_CK[2]#
32	VSS	VSS	DDR_DQ[63]	VSS	DDR_DM[7]	DDR_DQ[56]	DDR_DQ[43]	DDR_DQ[42]	DDR_DQS[5]	DDR_DQ[40]	DDR_ECC[3]	DDR_ECC[6]	DDR_DQS[8]	DDR_ECC[0]	DDR_CK[1]	DDR_ECC[4]	DDR_CK[2]
33	VSS	VSS	VSS	VSS	NC47	DDR_DQ[57]	DDR_DQ[60]	DDR_DQ[47]	DDR_DQS[5]#	DDR_DQ[41]	DDR_DQ[44]	DDR_ECC[7]	DDR_DQS[8]#	DDR_ECC[1]	DDR_CK[1]#	DDR_DRVCRES	DDR_DQ[26]

Table 48-37. EP80579 Ball Map (top view, right side) (Sheet 1 of 2)

T	R	P	N	M	L	K	J	H	G	F	E	D	C	B	A	
SIU_DCD2#	SIU_CTS1#	GP26_SATA0GP	CLK14	RCIN#	GP23_IRQ28	LFRAME#	SERIRQ	GP3_PIRQF#	GP5_PIRQH#	RTCX1	VSS	GP2_PIRQE#	VSS	VSS		1
SIU_CTS2#	SIU_TXD1	SPKR	PCICLK	SMI_OUT#	LDRQ[0]#	GP18_IRQ36	VSS	GPIO[12]	V_SEL	RTCX2	VSS	VSS	VSS	VSS	VSS	2
SIU_TXD2	SIU_RXD1	GP40_IRQ35	STPCLK_OUT#	VSS	LAD[2]	GP17_IRQ25	SATALED#	GP16_IRQ24	RSMRST#	Reserved11	VSS	VSS	VSS	PME#	VSS	3
SIU_RXD2	VSS	GP33_IRQ33	GP21_IRQ27	VCC33	LAD[3]	VCC50_SUS	GPIO[7]	VCC33	GP29_SATA1GP	WDT_TOUT#	GP11_SMBALERT#	GPIO[9]	VSS	GPIO[10]	GPIO[15]	4
SIU_DCD1#	NMI	CPUSLP_OUT#	GP30_IRQ31	VCCPRTC	LAD[1]	VSS	GPIO[1]	VSS	GPIO[14]	VSS	VCC33	GPIO[8]	SMBCLK	SMLINK[1]	PLTRST#	5
PROCHOT#	VCC33	GP41_LDRQ[1]#	LAD[0]	VSS	GPIO[0]	PWROK	NC_SUS_TWO	VCC33	SMBDATA	VSS	VSS	Reserved10	VSS	SUS_STAT#	SLP_S3#	6
GP34_IRQ34	VSS	GP20_IRQ26	GP4_PIRQG#	CPUPWRGD_OUT	RTEST#	GP24_IRQ29	GP25_IRQ38	SMLINK[0]	PCIRST#	VSS	SUSCLK	SYS_RESET#	SLP_S4#	VCCUSB12	VSS	7
NC7	VCC	GP19_IRQ37	GPIO[13]	BSEL	INTRUDER#	GP27_IRQ39	GP28_IRQ30	PWRBTN#	VSS	VSS	VSS	RI#	VCCUSB12	NC21	NC22	8
VSS	VCC33	VCC50	GPIO[6]	VCCSUS1	VSS	VCC50	VSS	SLP_S5#	VSS	VSS	VCCUSB12	VSS	VSS	USB_RBIASp	USB_RBIASn	9
VSS	VCC	VSS	VCC	VCCSUS1	VCC	VSS	VCCSUS1	VCCSUS1	VCCUSB12	VCCPSUS	VSS	VSS	VSS	VCCUSB12	VSS	10
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPSUS	VSS	VCCPSUS	VCCUSB12	VSS	CLK48	VCCUSB12	USBp0	USBn0	11
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC1P2_USBSUS	VCCABG3P3_USB	VSS	NC20	OC[0]#	OC[1]#	VSS	USBp1	USBn1	12
VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCAUSB12	VSS	SATA_CLKREFp	VCCAUSB12	VSS	VCCAUSB12	VCCSATA33	VCCARX	VSS	13
VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCSATA	VCCSATA	SATA_CLKREFn	VCCARX	SATA_TXn0	SATA_TXp0	VCCARX	SATA_RXn0	SATA_RXp0	14
VCC	VSS	VCC	VSS	VCC	VSS	VCCVC	VSS	VCCASATABG3P3	VCCABGP033	SATA_RBIAS	SATA_TXn1	SATA_TXp1	VCCAPLL	SATA_RXn1	SATA_RXp1	15
VSS	VCC	VSS	VCC	VSS	VCCVC	VSS	VCCVC	VSS	VCCAPE	SATA_RBIAS#	VCCATX	VSS	VCCATX	VCCATX	VSS	16
VCC	VSS	VCC	VSS	VCCVC	VSS	VCCVC	VSS	VSS	VSS	VCCRPE	PEA0_Tp[6]	PEA0_Tn[6]	VSS	PEA0_Rp[6]	PEA0_Rn[6]	17
VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	NC13	NC15	VCCAPE0PLL12	PEA0_Tn[7]	PEA0_Tp[7]	VCCRPE	PEA0_Rn[7]	PEA0_Rp[7]	18
VCCAHPLL	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	NC14	NC12	VCCAPE	PEA0_Tp[4]	PEA0_Tn[4]	VCCAPE	PEA0_Rp[4]	PEA0_Rn[4]	19
VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	NC16	VCCAPE	VCCAPE0PLL12	PEA0_Tn[5]	PEA0_Tp[5]	VSS	PEA0_Rn[5]	PEA0_Rp[5]	20

Table 48-37.EP80579 Ball Map (top view, right side) (Sheet 2 of 2)

T	R	P	N	M	L	K	J	H	G	F	E	D	C	B	A	
VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	Reserved9	VCCAPE	PEA_RCOMPO	VCCRPE	VSS	VCCAPE	VSS	21
VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VCCRPE	PEA_CLKp	PEA_CLKn	VCCAPE	PEA_ICOMPI	PEA_ICOMPO	22
VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCAPE	VCCRPE	VCCRPE	VSS	VCCRPE	VSS	23
VCCVC	VCCVC	VSS	VCCVC	VSS	VCCVC	VSS	VCCVC	VCCVC	VCCA[1]	VCCA[2]	PEA0_Tp[3]	PEA0_Tn[3]	VCCAPE	PEA0_Rp[3]	PEA0_Rn[3]	24
VCC18	VSS	VCCVC	VCCTMP18	DDR_CKE[1]	VSS	NC48	NC51	VSS	VSS	VCCAPE	PEA0_Tn[2]	PEA0_Tp[2]	VSS	PEA0_Rn[2]	PEA0_Rp[2]	25
VSS	VSS	VSS	VSS	BPM1	DDR_ODT1	NC36	VCCVC	VCCVC	DDR_CK[5]	THERMDA	PEA0_Tp[1]	PEA0_Tn[1]	VCCRPE	PEA0_Rp[1]	PEA0_Rn[1]	26
DDR_DQ[31]	VCC18	NC42	DDR_DQ[17]	DDR_DQ[21]	NC52	NC37	DDR_CK[3]#	VCCVC	DDR_CK[5]#	THERMDC	PEA0_Tn[0]	PEA0_Tp[0]	VSS	PEA0_Rn[0]	PEA0_Rp[0]	27
DDR_DQ[18]	VSS	DDR_DM[2]	DDR_DQ[16]	DDR_DQ[20]	VSS	DDR_CS[1]#	DDR_CK[3]	DDR_CS[0]#	BPM3	NC18	NC17	NC19	VCCAPE	VSS	VCCAPE	28
DDR_DQ[23]	DDR_DQS[2]	VSS	VCC18	VSS	VTTDDR	VSS	VCC18	VSS	VSS	VTTDDR	NC34	NC54	VSS	Reserved17	VCCVC	29
DDR_DQ[22]	DDR_DQS[2]#	VCC18	VSS	VCC18	VSS	VCC18	VSS	DDR_DQ[7]	VCC18	VSS	NC50	NC38	CLKP100	VCCVC	VSS	30
DDR_DQ[30]	VSS	VSS	DDR_DQ[29]	DDR_DQ[10]	DDR_DQS[1]	DDR_DQ[8]	DDR_DQ[3]	DDR_DQ[6]	NC40	DDR_DM[0]	DDR_DQ[4]	DDR_CK[4]	NC35	CLKN100	VSS	31
DDR_DQS[3]#	DDR_DM[3]	DDR_DQ[25]	DDR_DQ[28]	DDR_DQ[15]	DDR_DQS[1]#	DDR_DQ[9]	DDR_DQ[2]	VSS	DDR_DQS[0]	DDR_DQ[0]	DDR_DQ[5]	DDR_CK[4]#	BPM2	VSS	VSS	32
DDR_DQS[3]	NC43	DDR_DQ[24]	DDR_DQ[11]	DDR_DQ[14]	NC41	DDR_DM[1]	DDR_DQ[13]	DDR_DQ[12]	DDR_DQS[0]#	DDR_DQ[1]	BPM0	VSS	VSS	VSS	VSS	33

Table 48-38 provides package trace length information.

Table 48-38. Package Trace Length (Sheet 1 of 13)

Customer Signal	Total length (mm)	Total length (Mils)
1588_PPS	15.960	628.330
1588_RX_SNAP	13.581	534.677
1588_TESTMODE_DATA	16.580	652.746
1588_TX_SNAP	13.032	513.059
A20GATE	12.984	511.170
AMMSSIG	16.360	644.111
ASMSSIG	14.539	572.386
BPM0	19.790	779.153
BPM1	7.665	301.752
BPM2	23.372	920.163
BPM3	15.018	591.244
BPM3_IN	10.640	418.891
BPM4_PRDY_OUT	9.977	392.789
BPM5_PREQ_IN	7.131	280.761
BSEL	4.774	187.963
CLK14	13.402	527.633
CLK48	10.313	406.043
CLKN100	28.039	1103.907
CLKP100	28.040	1103.919
CN0RXD	9.886	389.208
CN0TXD	5.437	214.060
CN0TXEN	6.089	239.722
CN1RXD	15.780	621.272
CN1TXD	13.976	550.234
CN1TXEN	9.552	376.073
CPUPWRGD_OUT	6.051	238.228
CPUSLP_OUT#	8.152	320.942
DDR_A[0]	11.991	472.082
DDR_A[1]	12.520	492.928
DDR_A[10]	12.664	498.573
DDR_A[11]	10.166	400.237
DDR_A[12]	8.497	334.547
DDR_A[13]	13.348	525.503
DDR_A[14]	7.905	311.224
DDR_A[2]	12.209	480.670
DDR_A[3]	12.519	492.866
DDR_A[4]	10.987	432.547
DDR_A[5]	11.173	439.863
DDR_A[6]	11.900	468.487

Table 48-38. Package Trace Length (Sheet 2 of 13)

DDR_A[7]	10.914	429.703
DDR_A[8]	10.953	431.225
DDR_A[9]	8.565	337.189
DDR_BA[0]	13.353	525.719
DDR_BA[1]	14.483	570.216
DDR_BA[2]	7.122	280.410
DDR_CAS#	12.640	497.644
DDR_CK[0]	21.164	833.246
DDR_CK[0]#	21.164	833.220
DDR_CK[1]	21.166	833.317
DDR_CK[1]#	21.165	833.283
DDR_CK[2]	21.160	833.058
DDR_CK[2]#	21.164	833.237
DDR_CK[3]	21.163	833.191
DDR_CK[3]#	21.163	833.205
DDR_CK[4]	21.164	833.220
DDR_CK[4]#	21.162	833.159
DDR_CK[5]	21.162	833.165
DDR_CK[5]#	21.163	833.189
DDR_CKE[0]	14.509	571.209
DDR_CKE[1]	6.765	266.353
DDR_CRES[0]	6.325	249.020
DDR_CRES[1]	9.819	386.589
DDR_CRES[2]	9.819	386.594
DDR_CS[0]#	17.620	693.688
DDR_CS[1]#	11.412	449.308
DDR_DM[0]	21.573	849.348
DDR_DM[1]	18.891	743.742
DDR_DM[2]	15.674	617.103
DDR_DM[3]	16.218	638.499
DDR_DM[4]	9.844	387.567
DDR_DM[5]	16.331	642.951
DDR_DM[6]	9.274	365.120
DDR_DM[7]	17.633	694.213
DDR_DM[8]	16.625	654.511
DDR_DQ[0]	21.574	849.363
DDR_DQ[1]	21.578	849.520
DDR_DQ[10]	18.891	743.726
DDR_DQ[11]	18.882	743.372
DDR_DQ[12]	18.891	743.744
DDR_DQ[13]	18.896	743.920
DDR_DQ[14]	18.896	743.955

Table 48-38. Package Trace Length (Sheet 3 of 13)

DDR_DQ[15]	18.869	742.854
DDR_DQ[16]	15.762	620.541
DDR_DQ[17]	15.764	620.616
DDR_DQ[18]	15.763	620.590
DDR_DQ[19]	15.766	620.689
DDR_DQ[2]	21.573	849.315
DDR_DQ[20]	15.759	620.444
DDR_DQ[21]	15.762	620.563
DDR_DQ[22]	15.763	620.577
DDR_DQ[23]	15.760	620.462
DDR_DQ[24]	16.207	638.063
DDR_DQ[25]	16.220	638.574
DDR_DQ[26]	16.207	638.061
DDR_DQ[27]	16.208	638.123
DDR_DQ[28]	16.219	638.530
DDR_DQ[29]	16.216	638.409
DDR_DQ[3]	21.576	849.445
DDR_DQ[30]	16.208	638.102
DDR_DQ[31]	16.214	638.328
DDR_DQ[32]	9.830	387.025
DDR_DQ[33]	9.846	387.648
DDR_DQ[34]	9.849	387.771
DDR_DQ[35]	9.844	387.570
DDR_DQ[36]	9.844	387.567
DDR_DQ[37]	9.841	387.439
DDR_DQ[38]	9.797	385.700
DDR_DQ[39]	9.844	387.577
DDR_DQ[4]	21.705	854.517
DDR_DQ[40]	16.363	644.215
DDR_DQ[41]	16.328	642.825
DDR_DQ[42]	16.329	642.874
DDR_DQ[43]	16.347	643.580
DDR_DQ[44]	16.332	642.987
DDR_DQ[45]	16.332	642.996
DDR_DQ[46]	16.314	642.268
DDR_DQ[47]	16.296	641.593
DDR_DQ[48]	9.275	365.140
DDR_DQ[49]	9.278	365.279
DDR_DQ[5]	21.641	852.025
DDR_DQ[50]	9.263	364.685
DDR_DQ[51]	9.263	364.683
DDR_DQ[52]	9.281	365.387

Table 48-38. Package Trace Length (Sheet 4 of 13)

DDR_DQ[53]	9.279	365.315
DDR_DQ[54]	9.286	365.578
DDR_DQ[55]	9.323	367.049
DDR_DQ[56]	17.637	694.380
DDR_DQ[57]	17.613	693.420
DDR_DQ[58]	17.621	693.739
DDR_DQ[59]	17.653	694.993
DDR_DQ[6]	21.578	849.508
DDR_DQ[60]	17.632	694.158
DDR_DQ[61]	17.634	694.259
DDR_DQ[62]	17.636	694.313
DDR_DQ[63]	17.630	694.105
DDR_DQ[7]	21.577	849.491
DDR_DQ[8]	18.878	743.244
DDR_DQ[9]	18.887	743.585
DDR_DQS[0]	21.573	849.330
DDR_DQS[0]#	21.579	849.555
DDR_DQS[1]	18.895	743.887
DDR_DQS[1]#	18.896	743.928
DDR_DQS[2]	15.763	620.607
DDR_DQS[2]#	15.762	620.552
DDR_DQS[3]	16.215	638.376
DDR_DQS[3]#	16.210	638.183
DDR_DQS[4]	9.851	387.826
DDR_DQS[4]#	9.852	387.865
DDR_DQS[5]	16.326	642.759
DDR_DQS[5]#	16.327	642.790
DDR_DQS[6]	9.282	365.449
DDR_DQS[6]#	9.282	365.450
DDR_DQS[7]	17.635	694.288
DDR_DQS[7]#	17.636	694.332
DDR_DQS[8]	16.623	654.465
DDR_DQS[8]#	16.622	654.409
DDR_DRVCRES	17.435	686.408
DDR_ECC[0]	16.624	654.476
DDR_ECC[1]	16.639	655.072
DDR_ECC[2]	16.632	654.806
DDR_ECC[3]	16.625	654.526
DDR_ECC[4]	16.623	654.455
DDR_ECC[5]	16.624	654.483
DDR_ECC[6]	16.618	654.260
DDR_ECC[7]	16.631	654.761

Table 48-38. Package Trace Length (Sheet 5 of 13)

DDR_ODT0	13.422	528.444
DDR_ODT1	9.022	355.190
DDR_RAS#	7.307	287.675
DDR_RCOMPX	5.341	210.280
DDR_SLEWCRES	7.752	305.192
DDR_WE#	13.589	535.005
EECS	6.423	252.856
EEDI	5.028	197.952
EEDO	7.934	312.350
EESK	3.712	146.150
EX_ADDR[0]	9.577	377.058
EX_ADDR[1]	8.329	327.909
EX_ADDR[10]	14.301	563.029
EX_ADDR[11]	5.763	226.882
EX_ADDR[12]	11.074	435.991
EX_ADDR[13]	7.193	283.205
EX_ADDR[14]	12.202	480.380
EX_ADDR[15]	11.131	438.227
EX_ADDR[16]	15.750	620.088
EX_ADDR[17]	7.988	314.474
EX_ADDR[18]	14.348	564.887
EX_ADDR[19]	6.211	244.535
EX_ADDR[2]	11.191	440.593
EX_ADDR[20]	8.508	334.979
EX_ADDR[21]	10.344	407.225
EX_ADDR[22]	15.206	598.648
EX_ADDR[23]	12.119	477.116
EX_ADDR[24]	14.198	558.982
EX_ADDR[3]	9.758	384.181
EX_ADDR[4]	13.241	521.283
EX_ADDR[5]	5.231	205.950
EX_ADDR[6]	14.716	579.354
EX_ADDR[7]	6.677	262.866
EX_ADDR[8]	15.010	590.942
EX_ADDR[9]	13.169	518.482
EX_ALE	13.633	536.744
EX_BE[0]#	13.966	549.828
EX_BE[1]#	13.394	527.324
EX_BURST	11.580	455.896
EX_CLK	7.766	305.734
EX_CS[0]#	10.246	403.396
EX_CS[1]#	12.869	506.664

Table 48-38. Package Trace Length (Sheet 6 of 13)

EX_CS[2]#	11.633	457.976
EX_CS[3]#	6.433	253.281
EX_CS[4]#	13.385	526.957
EX_CS[5]#	5.122	201.662
EX_CS[6]#	12.570	494.893
EX_CS[7]#	9.169	361.002
EX_DATA[0]	12.549	494.074
EX_DATA[1]	4.370	172.049
EX_DATA[10]	11.464	451.329
EX_DATA[11]	6.517	256.557
EX_DATA[12]	10.127	398.700
EX_DATA[13]	5.880	231.493
EX_DATA[14]	9.244	363.939
EX_DATA[15]	12.536	493.546
EX_DATA[2]	9.659	380.268
EX_DATA[3]	6.423	252.858
EX_DATA[4]	8.525	335.641
EX_DATA[5]	5.612	220.957
EX_DATA[6]	11.127	438.090
EX_DATA[7]	8.135	320.272
EX_DATA[8]	12.701	500.035
EX_DATA[9]	10.201	401.600
EX_IOWAIT#	11.779	463.724
EX_PARITY[0]	10.170	400.413
EX_PARITY[1]	9.016	354.978
EX_RCOMPN	6.527	256.976
EX_RCOMPP	8.848	348.345
EX_RD#	8.364	329.302
EX_RDY[0]#	10.024	394.657
EX_RDY[1]#	4.821	189.808
EX_RDY[2]#	7.015	276.185
EX_RDY[3]#	5.475	215.564
Reserved20	7.684	302.530
NC58	5.872	231.173
NC59	2.846	112.054
EX_WR#	10.965	431.704
GBE_AUX_PWR_GOOD	12.179	479.487
GBE_PME_WAKE	2.435	95.851
GBE_RCOMPN	3.108	122.376
GBE_RCOMPP	4.314	169.835
GBE_REFCLK	12.170	479.131
GBE_REFCLK_RMII	9.952	391.815

Table 48-38. Package Trace Length (Sheet 7 of 13)

GBE0_RxCLK	12.511	492.561
GBE0_RxCTL	12.513	492.630
GBE0_RXDATA[0]	12.512	492.611
GBE0_RXDATA[1]	12.515	492.720
GBE0_RXDATA[2]	12.507	492.398
GBE0_RXDATA[3]	12.550	494.094
GBE0_TxCLK	11.522	453.608
GBE0_TxCTL	11.550	454.706
GBE0_TXDATA[0]	11.537	454.199
GBE0_TXDATA[1]	11.546	454.566
GBE0_TXDATA[2]	11.543	454.441
GBE0_TXDATA[3]	11.550	454.718
GBE1_RxCLK	13.836	544.735
GBE1_RxCTL	13.706	539.617
GBE1_RXDATA[0]	13.774	542.284
GBE1_RXDATA[1]	13.807	543.568
GBE1_RXDATA[2]	13.830	544.505
GBE1_RXDATA[3]	12.812	504.417
GBE1_TxCLK	11.354	447.015
GBE1_TxCTL	11.318	445.590
GBE1_TXDATA[0]	11.575	455.719
GBE1_TXDATA[1]	11.321	445.714
GBE1_TXDATA[2]	11.317	445.570
GBE1_TXDATA[3]	11.385	448.228
GBE2_RxCLK	14.349	564.924
GBE2_RxCTL	14.448	568.828
GBE2_RXDATA[0]	14.454	569.038
GBE2_RXDATA[1]	14.397	566.791
GBE2_RXDATA[2]	14.364	565.506
GBE2_RXDATA[3]	14.427	568.002
GBE2_TxCLK	9.814	386.392
GBE2_TxCTL	9.829	386.978
GBE2_TXDATA[0]	9.860	388.186
GBE2_TXDATA[1]	9.844	387.569
GBE2_TXDATA[2]	9.821	386.652
GBE2_TXDATA[3]	9.816	386.449
GP11_SMBALERT#	13.735	540.749
GP16_IRQ24	12.667	498.714
GP17_IRQ25	11.633	458.005
GP18_IRQ36	12.895	507.672
GP19_IRQ37	4.294	169.053
GP2_PIRQE#	21.089	830.274

Table 48-38. Package Trace Length (Sheet 8 of 13)

GP20_IRQ26	5.504	216.709
GP21_IRQ27	9.363	368.634
GP23_IRQ28	13.423	528.461
GP24_IRQ29	6.180	243.295
GP25_IRQ38	7.118	280.246
GP26_SATA0GP	13.028	512.932
GP27_IRQ39	5.441	214.229
GP28_IRQ30	6.145	241.948
GP29_SATA1GP	13.651	537.454
GP3_PIRQF#	15.305	602.548
GP30_IRQ31	8.623	339.477
GP31_IRQ32	12.325	485.234
GP33_IRQ33	9.933	391.067
GP34_IRQ34	5.340	210.233
GP4_PIRQG#	5.858	230.617
GP40_IRQ35	10.978	432.192
GP41_LDRQ[1]#	6.898	271.569
GP5_PIRQH#	15.920	626.760
GPIO[0]	9.010	354.741
GPIO[1]	10.515	413.962
GPIO[10]	16.548	651.515
GPIO[12]	13.810	543.711
GPIO[13]	4.829	190.126
GPIO[14]	9.831	387.039
GPIO[15]	16.505	649.793
GPIO[48]	7.488	294.791
GPIO[6]	3.529	138.922
GPIO[7]	11.371	447.666
GPIO[8]	13.764	541.881
GPIO[9]	14.464	569.455
IERR#	12.024	473.375
INIT33V_OUT#	14.076	554.186
INTRUDER#	4.825	189.978
LAD[0]	7.962	313.446
LAD[1]	10.784	424.578
LAD[2]	10.981	432.324
LAD[3]	10.887	428.623
LDRQ[0]#	12.223	481.210
LFRAME#	13.960	549.624
MDC	4.939	194.456
MDIO	6.926	272.661
NC56	8.969	353.099

Table 48-38. Package Trace Length (Sheet 9 of 13)

NC57	5.263	207.203
NMI	7.702	303.246
OC[0]#	9.444	371.817
OC[1]#	9.652	379.993
PCICLK	12.045	474.210
PCIRST#	8.107	319.175
PE_HPINTR#	11.288	444.427
PEA_CLKn	12.821	504.759
PEA_CLKp	12.820	504.708
PEA_ICOMPI	14.484	570.217
PEA_ICOMPO	17.850	702.754
PEA_RCOMPO	10.914	429.683
PEA0_Rn[0]	17.987	708.157
PEA0_Rn[1]	19.005	748.238
PEA0_Rn[2]	17.869	703.509
PEA0_Rn[3]	16.952	667.417
PEA0_Rn[4]	13.150	517.714
PEA0_Rn[5]	14.652	576.834
PEA0_Rn[6]	12.543	493.836
PEA0_Rn[7]	12.432	489.463
PEA0_Rp[0]	17.986	708.127
PEA0_Rp[1]	19.006	748.251
PEA0_Rp[2]	17.871	703.580
PEA0_Rp[3]	16.953	667.460
PEA0_Rp[4]	13.150	517.721
PEA0_Rp[5]	14.652	576.843
PEA0_Rp[6]	12.545	493.878
PEA0_Rp[7]	12.433	489.489
PEA0_Tn[0]	15.850	624.018
PEA0_Tn[1]	15.885	625.394
PEA0_Tn[2]	15.158	596.780
PEA0_Tn[3]	14.917	587.283
PEA0_Tn[4]	11.376	447.854
PEA0_Tn[5]	12.011	472.889
PEA0_Tn[6]	10.478	412.523
PEA0_Tn[7]	10.320	406.280
PEA0_Tp[0]	15.852	624.096
PEA0_Tp[1]	15.886	625.441
PEA0_Tp[2]	15.155	596.668
PEA0_Tp[3]	14.920	587.387
PEA0_Tp[4]	11.374	447.781
PEA0_Tp[5]	12.015	473.021

Table 48-38. Package Trace Length (Sheet 10 of 13)

PEA0_Tp[6]	10.473	412.327
PEA0_Tp[7]	10.320	406.299
PLTRST#	14.629	575.934
PME#	17.160	675.587
PROCHOT#	6.607	260.129
PWRBTN#	5.979	235.412
PWRGD	6.614	260.394
PWROK	8.173	321.784
RCIN#	13.116	516.392
Reserved18	3.466	136.458
Reserved19	2.693	106.039
RI#	9.836	387.243
RSMRST#	13.264	522.220
RSTIN#	13.004	511.971
RTCX1	15.716	618.739
RTCX2	14.528	571.956
RTEST#	6.026	237.230
Rx_CLK0	7.781	306.336
Rx_CLK1	12.076	475.418
Rx_CLK2	10.037	395.165
Rx_DATA_IN0	11.532	454.030
Rx_DATA_IN1	13.708	539.697
Rx_DATA_IN2	6.966	274.245
Rx_FRAME0	11.473	451.709
Rx_FRAME1	5.124	201.729
Rx_FRAME2	11.077	436.117
SATA_CLKREFn	6.887	271.137
SATA_CLKREFp	6.887	271.122
SATA_RBIAS	7.244	285.180
SATA_RBIAS#	7.240	285.043
SATA_RXn0	11.368	447.551
SATA_RXn1	11.879	467.668
SATA_RXp0	11.367	447.539
SATA_RXp1	11.878	467.618
SATA_TXn0	8.527	335.728
SATA_TXn1	8.737	343.980
SATA_TXp0	8.528	335.767
SATA_TXp1	8.737	343.988
SATALED#	12.290	483.847
SERIRQ	14.808	582.997
SIU_CTS1#	12.658	498.346
SIU_CTS2#	11.522	453.641

Table 48-38. Package Trace Length (Sheet 11 of 13)

SIU_DCD1#	8.662	341.015
SIU_DCD2#	12.771	502.813
SIU_DSR1#	8.047	316.821
SIU_DSR2#	9.299	366.085
SIU_DTR1#	5.291	208.319
SIU_DTR2#	13.306	523.852
SIU_RI1#	10.683	420.571
SIU_RI2#	13.780	542.504
SIU_RTS1#	6.519	256.663
SIU_RTS2#	10.021	394.544
SIU_RXD1	10.428	410.555
SIU_RXD2	9.166	360.885
SIU_TXD1	11.680	459.861
SIU_TXD2	10.817	425.862
SLP_S3#	13.629	536.592
SLP_S4#	12.191	479.976
SLP_S5#	5.919	233.020
SMBCLK	14.262	561.513
SMBDATA	8.956	352.596
SMBSCL	8.225	323.817
SMBSDA	12.143	478.077
SMI_OUT#	12.124	477.316
SMLINK[0]	7.203	283.596
SMLINK[1]	15.500	610.226
SPI_CS#	12.377	487.297
SPI_MISO	13.590	535.039
SPI_MOSI	4.715	185.648
SPI_SCLK	11.543	454.435
SPKR	11.897	468.375
SSP_EXTCLK	11.658	458.963
SSP_RXD	6.956	273.865
SSP_SCLK	13.584	534.803
SSP_SFRM	11.775	463.580
SSP_TXD	14.865	585.241
STPCLK_OUT#	11.067	435.700
SUS_STAT#	12.854	506.066
SUSCLK	10.910	429.537
SYS_PWR_OK	8.287	326.244
SYS_RESET#	10.851	427.198
TCK	9.399	370.022
TDI	12.463	490.658
TDO	16.247	639.650

Table 48-38. Package Trace Length (Sheet 12 of 13)

THERMDA	15.663	616.660
THERMDC	21.693	854.044
THRMTRIP#	10.177	400.682
TMS	9.491	373.675
TRST#	5.699	224.384
Tx_CLK0	9.120	359.056
Tx_CLK1	12.360	486.597
Tx_CLK2	8.721	343.346
Tx_DATA_OUT0	12.435	489.561
Tx_DATA_OUT1	3.229	127.113
Tx_DATA_OUT2	12.124	477.319
Tx_FRAME0	6.446	253.781
Tx_FRAME1	13.373	526.502
Tx_FRAME2	7.880	310.248
UART_CLK	4.251	167.356
USB_RBIASn	12.848	505.813
USB_RBIASp	11.917	469.164
USBn0	12.052	474.470
USBn1	12.031	473.662
USBp0	12.054	474.563
USBp1	12.034	473.779
V_SEL	14.260	561.437
VCC	717.956	28266.002
VCC	10.440	411.038
VCC18	38.611	1520.102
VCC1P2_USBSUS	4.633	182.385
VCC25	4.056	159.675
VCC33	23.406	921.500
VCC50	33.496	1318.736
VCC50_SUS	11.967	471.138
VCCA[1]	13.919	547.982
VCCA[2]	13.440	529.123
VCCABG3P3_USB	4.121	162.225
VCCABGP033	6.969	274.374
VCCAHPLL	2.351	92.569
VCCAPE	10.158	399.930
VCCAPE0PLL12	7.954	313.154
VCCAPE0PLL12	9.638	379.466
VCCAPLL	11.901	468.528
VCCARX	0.784	30.856
VCCASATABG3P3	3.618	142.431
VCCATX	0.784	30.856

Table 48-38. Package Trace Length (Sheet 13 of 13)

VCCAUSB12	1.045	41.142
VCCGBE33	4.637	182.555
VCCGBEPSUS	8.735	343.885
VCCPRTC	7.487	294.763
VCCPSUS	4.773	187.903
VCCRPE	7.535	296.639
VCCSATA	2.491	98.056
VCCSATA33	11.246	442.738
VCCSUS1	15.545	612.007
VCCSUS25	4.409	173.599
VCCTMP18	8.621	339.413
VCCUSB12	4.232	166.620
VCCVC	10.622	418.189
VCCVC	204.574	8054.101
VRMPWRGD	5.588	219.986
VSS	266.015	10473.015
VSSA	0.784	30.857
VTTDDR	7.470	294.093
WDT_TOUT#	14.519	571.632

§ §

49.0 Electrical Specifications

This chapter provides electrical specifications for the EP80579 interfaces.

The information contained in this chapter is divided into the following sections:

Section Title	Section Link
Absolute Maximum Ratings	Section 49.1
Input and I/O Pin Undershoot and Overshoot	Section 49.1.1
Power Characteristics	Section 49.2
Power Supply Requirements	Section 49.2.1
Clocks	Section 49.3
Power and Reset Sequencing	Section 49.4
AC/DC Characteristics	Section 49.5
Power Management	Section 49.5.1
DDR2	Section 49.5.2
PCI Express*	Section 49.5.3
Serial ATA (SATA)	Section 49.5.4
Universal Serial Bus (USB)	Section 49.5.5
System Management Bus (SMBus)	Section 49.5.6
Universal Asynchronous Receiver/Transmitter (UART)	Section 49.5.7
Serial Peripheral Interface (SPI)	Section 49.5.8
Low Pin Count (LPC)	Section 49.5.9
General Purpose I/O (GPIO)	Section 49.5.10
IICH Interrupt Signal	Section 49.5.11
Real Time Clock	Section 49.5.12
Gigabit Ethernet (GbE: RMII, RGMII, MDIO, EEPROM)	Section 49.5.13
Time Division Multiplex (TDM)	Section 49.5.14
Local Expansion Bus (LEB)	Section 49.5.15
Controller Access Network (CAN)	Section 49.5.16
Sync Serial Port (SSP)	Section 49.5.17
IEEE 1588-2008 Hardware Assist Interface	Section 49.5.18
IICH Miscellaneous Signals (Misc. IICH Signals)	Section 49.5.19
Clock Resource Unit (CRU)	Section 49.5.20
Sideband Miscellaneous Signals	Section 49.5.21
IMCH Reset	Section 49.5.22
JTAG	Section 49.5.23

49.1 Absolute Maximum Ratings

Absolute maximum ratings are limits beyond which may cause permanent damage to the device or affect device reliability. Stressing the device beyond the “absolute maximum ratings” may cause permanent damage.

These are stress ratings only. Operate all supplies within the stated nominal operating voltage described in Table 49-6, “Operating Conditions Power Supply Rails” on page 1823.

Absolute maximum ratings are not the normal operating conditions of the device. All voltages are specified with respect to GND unless otherwise specified.

Table 49-1. Absolute Maximum Ratings

Parameter	Maximum Rating	Notes
Operating Temperature	0° C to 70° C	
Operating Temperature (Industrial)	-40° C to 85° C	
Storage Temperature	-10° C to 45° C	1
Supply Voltage IA-32 core 600 MHz SKU (1.0V nominal)	1.03V	
Supply Voltage IA-32 core 1066/1200 MHz SKU (1.3V nominal)	1.326V	
Supply Voltage DDR2 (1.8V nominal)	1.89V	
Supply Voltage I/O (1.2V nominal)	1.26V	
Supply Voltage I/O (2.5V nominal)	2.62V	
Supply Voltage I/O (3.3V nominal)	3.46V	
Supply Voltage I/O (5.0V nominal)	5.25V	

Note:

1. The component storage temperature range (Pre-Board Assembly) is -45°C to 75°C for short-term exposure and -10°C to 45°C for sustained exposure. Consistent with industry practice, Intel does not specify storage conditions beyond component level (e.g. board or system level). In addition to this storage temperature specification, compliance to the latest IPC/JEDEC J-STD-033B.1 joint industry standard is required for all Surface Mount Devices (SMDs). This document governs handling, packing, shipping and use of moisture/reflow sensitive SMDs.

49.1.1 Input and I/O Pin Undershoot and Overshoot Specifications

The following tables provide Input and I/O pin undershoot and overshoot specifications:

- Table 49-2: applies to PCI, JTAG, SATA, CRU (15 pins) Input and I/O pin groups
- Table 49-3: applies to DDR2 Input and I/O pin group
- Table 49-4: applies to GbE Input and I/O pin group
- Table 49-5: applies to Input and I/O pin groups RTC, SPI, USB, TDM, LEB, CAN, SSP, IEEE 1588-2008, ICH Miscellaneous, IICH, SMBus, UART, LPC, GPIO, Sideband Miscellaneous, IMCH, PMI, and Miscellaneous

Table 49-2. Undershoot and Overshoot for PCI, JTAG, SATA and CRU Signal Groups

VUS[a]	VOS[b]	PW[c]	Signal Group[d]
-0.35	1.35	2.00E-08	• Table 48-9, "PCI Express Interface Signals" on page 1741; • Table 48-27, "JTAG Interface Signals" on page 1769, • Table 48-17, "Serial ATA Interface Signals" on page 1753, • Table 48-5, "Global Clock and Reset (CRU) Signals" on page 1736
-0.4	1.4	2.00E-08	
-0.45	1.45	2.00E-08	
-0.5	1.5	8.88E-09	
-0.55	1.55	3.66E-09	
-0.6	1.6	1.66E-09	
-0.65	1.65	6.64E-10	
-0.7	1.7	2.46E-10	
-0.75	1.75	1.28E-10	
-0.8	1.8	8.47E-11	

a. VUS (Voltage UnderShoot) - Values of Vpeak (peak undershoot voltage) for the maximum Pulse Width specified.
b. VOS (Voltage OverShoot) - Values of Vpeak (peak overshoot voltage) for the maximum Pulse Width specified.
c. PW (Pulse Width) - Maximum Pulse Width allowed for a given Vpeak value (undershoot or overshoot), such that the reliability specifications of the device are not exceeded.
d. Values provided in this table apply to Input and I/O pins in signal groups: PCI, JTAG, SATA, and CRU (15 pins). For more details, follow the table links provided.

Table 49-3. Undershoot and Overshoot for DDR2 Signal Group

VUS[a]	VOS[b]	PW[c]	Signal Group[d]
-0.2	2	1.25E-10	Table 48-8, "DDR2 Interface Signals" on page 1739
-0.3	2.1	1.25E-10	
-0.4	2.2	1.92E-09	
-0.5	2.3	6.22E-10	
-0.6	2.4	1.80E-11	
-0.7	2.5	3.67E-12	
-0.8	2.6	2.98E-12	
-0.9	2.7	2.73E-12	
-1	2.8	2.58E-12	
-1.1	2.9	2.08E-12	

a. VUS (Voltage UnderShoot) - Values of Vpeak (peak undershoot voltage) for the maximum Pulse Width specified.
b. VOS (Voltage OverShoot) - Values of Vpeak (peak overshoot voltage) for the maximum Pulse Width specified.
c. PW (Pulse Width) - Maximum Pulse Width allowed for a given Vpeak value (undershoot or overshoot), such that the reliability specifications of the device are not exceeded.
d. Values provided in this table apply to Input and I/O pins in the DDR2 signal group. For more details, follow the table link provided.

Table 49-4. Undershoot and Overshoot for GbE Signal Group

VUS[a]	VOS[b]	PW[c]	Signal Group[d]
-0.1	3.4	1.25E-09	Table 48-22, "Gigabit Ethernet Interface Signals" on page 1760
-0.2	3.5	1.25E-09	
-0.3	3.6	1.25E-09	
-0.4	3.7	6.04E-10	
-0.5	3.8	1.44E-10	
-0.6	3.9	3.41E-11	
-0.7	4	9.55E-12	
-0.8	4.1	4.97E-12	
-0.9	4.2	3.28E-12	
-1	4.3	2.97E-12	

a. VUS (Voltage UnderShoot) - Values of Vpeak (peak undershoot voltage) for the maximum Pulse Width specified.
b. VOS (Voltage OverShoot) - Values of Vpeak (peak overshoot voltage) for the maximum Pulse Width specified.
c. PW (Pulse Width) - Maximum Pulse Width allowed for a given Vpeak value (undershoot or overshoot), such that the reliability specifications of the device are not exceeded.
d. Values provided in this table apply to Input and I/O pins in the GbE signal group. For more details, follow the table link provided.

Table 49-5. Undershoot and Overshoot for RTC, SPI, USB, TDM, LEBus, CAN, SSP, IEEE 1588-2008, ICH Miscellaneous, IICH, SMBus, UART, LPC, GPIO, Sideband Miscellaneous, IMCH, PMI, and Miscellaneous Signal Groups

VUS[a]	VOS[b]	PW[c]	Signal Group[d]
-0.1	3.4	1.00E-08	Refer to Table 48-3, "Signal Pin Description References" on page 1735 for links to Input and I/O pin signal descriptions
-0.2	3.5	1.00E-08	
-0.3	3.6	1.00E-08	
-0.4	3.7	1.00E-08	
-0.5	3.8	2.92E-09	
-0.6	3.9	7.17E-10	
-0.7	4	1.77E-10	
-0.8	4.1	5.96E-11	
-0.9	4.2	3.31E-11	
-1	4.3	2.58E-11	

a. VUS (Voltage UnderShoot) - Values of Vpeak (peak undershoot voltage) for the maximum Pulse Width specified.
b. VOS (Voltage OverShoot) - Values of Vpeak (peak overshoot voltage) for the maximum Pulse Width specified.
c. PW (Pulse Width) - Maximum Pulse Width allowed for a given Vpeak value (undershoot or overshoot), such that the reliability specifications of the device are not exceeded.
d. Values provided in this table apply to Input and I/O pins in signal groups: RTC, SPI, USB, TDM, LEBus, CAN, SSP, IEEE 1588-2008, ICH Miscellaneous, IICH, SMBus, UART, LPC, GPIO, Sideband Miscellaneous, IMCH, PMI, and Miscellaneous. For more details, follow the table links provided.

49.2 Power Characteristics

49.2.1 Power Supply Requirements

This section describes areas related to the EP80579's power, including:

- Description of power supply requirements
- Power wells and platform expectations for connecting the power wells
- Power planes

Table 49-6 and Table 49-7 defines the power supply requirements for each power rail.

Note: Further details for the EP80579 power supply rails are provided in Section 6-3, "EP80579 Power Supply Pins"

Table 49-6. Operating Conditions Power Supply Rails (Sheet 1 of 2)

Well/Voltage Rail	Symbol	Supply Types	Tolerance +/-	Min	Nominal	Max	Unit	Notes
Core								
1.0 V (IA-32 core)	VCCVC	Core power, Freq @ 600 MHz	2%	0.98	1.00	1.02	V	
1.3 V (IA-32 core)	VCCVC	Core power, Freq @ 1066 MHz	2%	1.274	1.30	1.326	V	
1.3 V (IA-32 core)	VCCVC	Core power, Freq @ 1200 MHz	2%	1.274	1.30	1.326	V	
1.2 V	VCCA[2:1]	Core PLL Analog Power	5%	1.14	1.20	1.26	V	
I/O								
1.2 V (SOC logic, I/O)	VCC	CRU, CRU_PAD, DDR2, Expansion Bus, GBE, IMCH_PAD, MISC I/O, PCI, Express*, SATA core power	5%	1.14	1.2	1.26	V	
	VCCAHPLL	CRU Analog PLL power	5%	1.14	1.2	1.26	V	
	VCCAPE0PLL12	PCI Express* PLL digital power	5%	1.14	1.2	1.26	V	
	VCCAPE	PCI Express* Transmitter/Receiver analog power	5%	1.14	1.2	1.26	V	
	VCCRPE	PCI Express* Receiver digital power	5%	1.14	1.2	1.26	V	
	VCCAPLL	SATA Analog PLL power	5%	1.14	1.2	1.26	V	
	VCCARX	SATA Analog receiver power	5%	1.14	1.2	1.26	V	
	VCCATX	SATA Analog transmitter power	5%	1.14	1.2	1.26	V	
	VCCSATA	SATA power	5%	1.14	1.2	1.26	V	
	VCCAUSB12	USB Analog power	5%	1.14	1.2	1.26	V	
	VCCUSB12	USB Digital power	5%	1.14	1.2	1.26	V	
0.9 V (DDR2)	VTTDDR	DDR2 reference	VTTDDR derived from VCC18, refer to Section 49.5.2.2, "DDR2 DC Characteristics"					
1.8 V	VCC18	DDR2 I/O power/ Random Number Generator power	5%	1.71	1.8	1.89	V	

Table 49-6. Operating Conditions Power Supply Rails (Sheet 2 of 2)

Well/Voltage Rail	Symbol	Supply Types	Tolerance +/-	Min	Nominal	Max	Unit	Notes
I/O Continued								
	VCCTMP18	Thermal Sensor Power	5%	1.71	1.8	1.89	V	
2.5 V	VCC25	GBE 2.5V power	5%	2.375	2.5	2.625	V	
3.3 V	VCC33	CRU_PAD, Expansion Bus, IMCH_PAD, MISC I/O power	5%	3.135	3.3	3.465	V	
	VCCABGP033	PCI Express* Bandgap analog power	5%	3.135	3.3	3.465	V	
	VCCGBE33	GBE 3.3 V power	5%	3.135	3.3	3.465	V	
	VCCSATA33	SATA power	5%	3.135	3.3	3.465	V	
	VCCABG3P3_USB	USB2 Analog bandgap power	5%	3.135	3.3	3.465	V	
	VCCASATABG3P3	SATA Analog bandgap power	5%	3.135	3.3	3.465	V	
5 V	VCC50	5V Power Supply	5%	4.75	5.0	5.25	V	
Suspend								
1.2 V sustain supply	VCC1P2_USBSUS	USB, 1.2 V sustain power	5%	1.14	1.2	1.26	V	
	VCCSUS1	GBE and IICH sustain power	5%	1.14	1.2	1.26		
2.5 V sustain supply	VCCSUS25	2.5 V sustain power	5%	2.375	2.5	2.625	V	
3.3 V sustain supply	VCCPSUS	IICH, USB sustain power, I/O logic	5%	3.135	3.3	3.465	V	
	VCCGBEPSUS	GBE 3.3 V Sustain power, I/O logic	5%	3.135	3.3	3.465	V	
5 V sustain supply	VCC50_SUS	5V sustain well	5%	4.75	5.0	5.25	V	
RTC								
3.3 VRTC	VCCPRTC	Real Time Clock power		2.0	3.3	3.465	V	1

Note:

1. 3.3 V power supply for the RTC (this supply can drop to 2.0 V if all other planes are shut off). This power is not expected to be shut off unless the RTC battery is removed or drained.

Table 49-7. Maximum Supply Current Embedded SKU (Sheet 1 of 2)

Well/ Voltage Rail	Symbol	Supply Types	SKU ID			Unit
			2, 8	4	6	
IA-32 core						
1.0 V (IA-32 core)	ICCVC	Max Supply Current on VCCVC	3.4	-	-	A
1.3 V (IA-32 core)	ICCVC	Max Supply Current on VCCVC	-	8.4	11.4	A
I/O						
1.2 V (SOC logic, I/O)	ICC1_2	Max Supply Current on VCC - SOC/SATA/ USB	9			A
	ICC_APE	Max Supply Current, PCI Express* Tx/Rx Analog power 1.2 V on VCCAPE	1			A
	ICC_SATA	Max Supply Current, SATA 1.2 V on VCCARX	0.20			A
		Max Supply Current, SATA 1.2 V on VCCATX	0.20			A
	ICC_APLL	Max Supply Current, SATA PLL 1.2 V on VCCAPLL	0.08			A
	ICC_PE0PLL12	Max Supply Current, PCI Express* PLL digital power 1.2 V on VCCAPE0PLL12	0.08			A
	ICC_CRU_PLL	Max Supply Current, CRU PLL 1.2 V on VCCAHPLL	0.06			A
	ICC_USB_PLL	Max Supply Current, USB PLL 1.2 V on VCCAUSB12	0.02			A
0.9 V (DDR2)	ITT_DDR2	Max Supply Current, DDR2 Termination on VTTDDR	0.95			A
1.8 V	ICC18_DDR2	Max Supply Current, DDR2 on VCC18	2.5			A
1.8 V	ICC18	Max Supply Current for Misc 1.8V on VCC18	0.3			A
2.5 V	ICC_GBE	Max Supply Current, GbE 2.5 V on VCC25	0.20			A
3.3 V	ICC_3_3	Max Supply Current, 3.3 V I/O on VCC33	1.0			A
	ICC_ABGP033	Max Supply Current, PCI Express* bandgap power 3.3 V on VCCABGP033	0.04			A

Table 49-7. Maximum Supply Current Embedded SKU (Sheet 2 of 2)

Well/ Voltage Rail	Symbol	Supply Types	SKU ID			Unit
			2, 8	4	6	
	ICC_ABG3P3USB	Max Supply Current, USB bandgap power 3.3 V on VCCABG3P3_USB	0.02			A
	ICC_ASATABG3P3	Max Supply Current, SATA AFE bandgap power 3.3 V on VCCASATABG3P3	0.04			A
5 V	ICC_5	Max Supply Current, 5 V on VCC50	0.10			A
Suspend						
1.2 V sustain supply	ICC_1SUS	Max Supply Current, 1.2 V Suspend Well on VCCSUS1	0.70			A
2.5 V sustain supply	ICC_25SUS	2.5 V Suspend Well Max Supply Current on VCCSUS25	0.10			A
3.3 V sustain supply	ICC_33SUS	Max Supply Current, 3.3 V Suspend Well on VCCPSUS	0.10			A
5 V sustain supply	ICC_5SUS	Max Supply Current, 5 V Suspend Well on VCC50_SUS	0.01			A
RTC						
3.3 VRTC	ICCRTC[1]	RTC Current, on VCCPRTC	6			µA

Notes:

1. ICCRTC nominal data is taken with VccPRTC at 3.0V while the system is under battery power and at room temperature. This is shown to provide an estimated battery life. Maximum value of ICCRTC is 2mA while operating under full power.

Table 49-8. Maximum Supply Current Accelerated SKU (Sheet 1 of 2)

Well/ Voltage Rail	Symbol	Supply Types	SKU ID			Unit
			1	3,7	5	
IA-32 core						
1.0 V (IA-32 core)	ICCVC	Max Supply Current on VCCVC	3.4	-	-	A
1.3 V (IA-32 core)	ICCVC	Max Supply Current on VCCVC	-	8.4	11.4	A
I/O						
1.2 V (SOC logic, I/O)	ICC1_2	Max Supply Current on VCC - SOC/SATA/ USB	11	12	12	A
	ICC_APE	Max Supply Current, PCI Express* Tx/Rx Analog power 1.2V on VCCAPE	1			A
	ICC_SATA	Max Supply Current, SATA 1.2V on VCCARX	0.20			A
		Max Supply Current, SATA 1.2V on VCCATX	0.20			A
	ICC_APLL	Max Supply Current, SATA PLL 1.2V on VCCAPLL	0.08			A
	ICC_PE0PLL12	Max Supply Current, PCI Express* PLL digital power 1.2V on VCCAPE0PLL12	0.08			A
	ICC_CRU_PLL	Max Supply Current, CRU PLL 1.2V on VCCAHPLL	0.06			A
	ICC_USB_PLL	Max Supply Current, USB PLL 1.2V on VCCAUSB12	0.02			A
0.9 V (DDR2)	ITT_DDR2	Max Supply Current, DDR2 Termination on VTTDDR	0.95			A
1.8 V	ICC18_DDR2	Max Supply Current, DDR2 on VCC18	2.5			A
1.8 V	ICC18	Max Supply Current for Misc 1.8V on VCC18	0.3			A
2.5 V	ICC_GBE	Max Supply Current, GbE 2.5V on VCC25	0.20			A
3.3 V	ICC_3_3	Max Supply Current, 3.3V I/O on VCC33	1.0			A
	ICC_ABGP033	Max Supply Current, PCI Express* bandgap power 3.3V on VCCABGP033	0.04			A

Table 49-8. Maximum Supply Current Accelerated SKU (Sheet 2 of 2)

Well/ Voltage Rail	Symbol	Supply Types	SKU ID			Unit
			1	3,7	5	
	ICC_ABG3P3USB	Max Supply Current, USB bandgap power 3.3V on VCCABG3P3_USB	0.02			A
	ICC_ASATABG3P3	Max Supply Current, SATA AFE bandgap power 3.3V on VCCASATABG3P3	0.04			A
5 V	ICC_5	Max Supply Current, 5V on VCC50	0.10			A
Suspend						
1.2 V sustain supply	ICC_1SUS	Max Supply Current, 1.2V Suspend Well on VCCSUS1	0.70			A
2.5 V sustain supply	ICC_25SUS	2.5V Suspend Well Max Supply Current on VCCSUS25	0.10			A
3.3 V sustain supply	ICC_33SUS	Max Supply Current, 3.3V Suspend Well on VCCPSUS	0.10			A
5 V sustain supply	ICC_5SUS	Max Supply Current, 5V Suspend Well on VCC50_SUS	0.01			A
RTC						
3.3 VRTC	ICCRTC[1]	RTC Current, on VCCPRTC	6			µA

Notes:

1. ICCRTC nominal data is taken with VccPRTC at 3.0V while the system is under battery power and at room temperature. This is shown to provide an estimated battery life. Maximum value of ICCRTC is 2mA while operating under full power.

49.3 Clocks

49.3.1 External Clock Requirements

External clocks are supplied to the internal PLLs as the reference clock and to the other I/O devices. The EP80579 has internal PLLs that use the reference clock to generate the internal clocks. Table 49-9 shows the external clocks that must be supplied for the EP80579 and its external interfaces.

Table 49-9. Platform External/Internal Clock Interface

Clock Domain	Frequency	Source
DDR2	200/266/320/400 MHz	Internal
GbE	2.5, 25, 125 MHz	External
CRU Clock	100/133 MHz	External
PCI Express*	100 MHz	External
Serial ATA	100 MHz	External
USB2	48 MHz	External
UART	48/14.7456 MHz	External
LPC	33 MHz	External
Expansion Bus	33/80 MHz	External
CLK14	14.31818 MHz	External
TDM	512 kHz to 8.192 MHz	External
SSP	3.864 MHz	External
RTC	32.768 KHz	External
SUSCLK	32.768 KHz	Internal
SMBus	10-100 KHz	External
SPI	17.86 MHz	Internal

49.4 Power and Reset Sequencing

The details for power and reset sequencing are described in Chapter 6.0, “EP80579 Power Sequencing and Reset Sequence”

49.5 AC/DC Characteristics

49.5.1 Power Management

This section describes the electrical characteristics of the Power Management interface.

49.5.1.1 Power Management Signal List

Refer to Table 48-19, “Power Management Interface Signals” on page 1756 for the power management signal description.

49.5.1.1.1 Power Management DC Characteristics

Table 49-10. Power Management DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	2
V_{IL}	Input voltage low	-	-	-	0.8	V	2
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1.
2. VIH and VIL for PWRBTN# is guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-11. Power Management DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	3, 2, 3
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1.
2. VOH and VOL for PWRBTN# is guaranteed by design. These values are typical values seen for this process, but not measured during production testing.
3. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.1.2 Power Management AC Characteristics

The following table and diagrams illustrate the sequencing that occurs for supported power state transitions. For more detail, see Section 27.6, "Sleep States".

Table 49-12. Power Sequencing Signal Timings (Sheet 1 of 2)

Sym	Parameter	Min	Max	Units	Notes	Fig #
t204	VccSus supplies active to RSMRST# inactive	10	–	ms	7	49-1
t214	Vcc supplies active to PWROK, VRMPWRGD active	99	–	ms	11	49-1, 49-3
t215	Vcc active to STPCLK# and CPUSLP# inactive	–	50	ns	-	49-1, 49-3
t217	PWROK and VRMPWRGD/ CPU_VRD_PWR_GD active and SYS_RESET# inactive to SUS_STAT# inactive and Processor I/F signals latched to strap value.	32	38	RTCCLK	1	49-1, 49-3
t218	SUS_STAT# inactive to PLTRST# and PCIRST# inactive	2	3	RTCCLK		49-1, 49-3
t230	VccSus active to SLP_S5#, SLP_S4#, SLP_S3#, SUS_STAT#, PLTRST# and PCIRST#active	–	50	ns	-	49-1
t231 t232	RSMRST# inactive to SUSCLK running, SLP_S5# inactive	–	110	ms	2	49-1
t233	SLP_S5# inactive to SLP_S4# inactive	See Note 10			10	49-1
t234	SLP_S4# inactive to SLP_S3# inactive	1	2	RTCCLK	1	49-1
t271	S1 Wake Event to CPUSLP# inactive	1	25	PCICLK	8	49-2
	SUSCLK duty cycle	30	70	%	9	
t280	STPCLK# active to NSI Message	0		PCICLK	3	49-2, 49-3
t281	NSI Message to CPUSLP# active	60	63	PCICLK	8	49-2
t283	NSI Message to SUS_STAT# active	2		RTCCLK	1	49-3
t284	SUS_STAT# active to PLTRST#, PCIRST# active	7	17	RTCCLK	1	49-3
t287	PLTRST#, PCIRST# active to SLP_S3# active	1	2	RTCCLK	1	49-3
t289	SLP_S3# active to PWROK, VRMPWRGD/ CPU_VRD_PWR_GD inactive	0		ms	4	49-3
t291	SLP_S3# active to SLP_S4# active	1	2	RTCCLK	1	49-3
t294	PWROK, VRMPWRGD/ CPU_VRD_PWR_GD inactive to Vcc supplies inactive	20		ns	6	49-3
t295	SLP_S4# active to SLP_S5# active	1	2	RTCCLK	1, 5	49-3
t296	Wake Event to SLP_S5# inactive	1	10	RTCCLK	1	49-3

Notes:

1. These transitions are clocked off the internal RTC. 1 RTC clock is approximately 32 µs.
2. If there is no RTC battery in the system, so VCCPRTC and the VCCPSUS supplies come up together, the delay from RTCRST# and an inactive RSMRST# prior to SUSCLK toggling may be as much as 2.5 s.
3. STPCLK# assertion triggers the processor to send a stop grant acknowledge cycle.
4. The EP80579 has no maximum timing requirement for this transition. It is up to the system designer to determine if the SLP_S3#, SLP_S4# and SLP_S5# signals are used to control the power planes.
5. If the transition to S5 is due to Power Button Override, SLP_S3#, SLP_S4# and SLP_S5# are asserted together similar to timing t287 (PCIRST# active to SLP_S3# active).
6. VCC in diagram represents all core well supplies that are powered off in S3 state.
7. VCCSus in diagram represents all sustainable supplies as identified in the Suspend power well.
8. These transitions are clocked off the 33MHz PCICLK. 1 PCICLK is approximately 30ns.
9. SUSCLK is an output of the RTC generator (32.768 KHz), has a duty cycle that can be as low as 30% or as high as 70%.
10. The Min/Max times depend on the programming of the "SLP_S4# Minimum Assertion Width" and the "SLP_S4# Assertion Stretch Enable bits. Note that this does not apply for synchronous SMI's.
11. The relationship that the active edge of SYS_PWR_OK (platform signal connected to PWROK, PWRGD, and SYS_PWR_OK pins) has to the active edge of VRMPWRGD/ CPU_VRD_PWR_GD in Figure 6-1 must be observed.

Table 49-12. Power Sequencing Signal Timings (Sheet 2 of 2)

Sym	Parameter	Min	Max	Units	Notes	Fig #
t297	SLP_S5# inactive to SLP_S4# inactive	See Note 10			10	49-3
t298	SLP_S4# inactive to SLP_S3# inactive	1	2	RTCCLK	1	49-3
t299	S4 to SLP_S4# inactive	See Note 10			10	49-3
t300	S3 Wake Event to SLP_S3# inactive (S3 Wake)	0	Small as possible	ms	1	49-3
t301	CPUSLP# inactive to STPCLK# inactive	8		PCICLK	-	49-2

Notes:

1. These transitions are clocked off the internal RTC. 1 RTC clock is approximately 32 µs.
2. If there is no RTC battery in the system, so VCCPRTC and the VCCPSUS supplies come up together, the delay from RTCRST# and an inactive RSMRST# prior to SUSCLK toggling may be as much as 2.5 s.
3. STPCLK# assertion triggers the processor to send a stop grant acknowledge cycle.
4. The EP80579 has no maximum timing requirement for this transition. It is up to the system designer to determine if the SLP_S3#, SLP_S4# and SLP_S5# signals are used to control the power planes.
5. If the transition to S5 is due to Power Button Override, SLP_S3#, SLP_S4# and SLP_S5# are asserted together similar to timing t287 (PCIRST# active to SLP_S3# active).
6. VCC in diagram represents all core well supplies that are powered off in S3 state.
7. VCCSus in diagram represents all sustainable supplies as identified in the Suspend power well.
8. These transitions are clocked off the 33MHz PCICLK. 1 PCICLK is approximately 30ns.
9. SUSCLK is an output of the RTC generator (32.768 KHz), has a duty cycle that can be as low as 30% or as high as 70%.
10. The Min/Max times depend on the programming of the “SLP_S4# Minimum Assertion Width” and the “SLP_S4# Assertion Stretch Enable bits. Note that this does not apply for synchronous SMI’s.
11. The relationship that the active edge of SYS_PWR_OK (platform signal connected to PWROK, PWRGD, and SYS_PWR_OK pins) has to the active edge of VRMPWRGD/ CPU_VRD_PWR_GD in Figure 6-1 must be observed.

49.5.1.2.1 Power Management Timing Diagrams

Figure 49-1. G3 (Mechanical Off) to S0 Timings

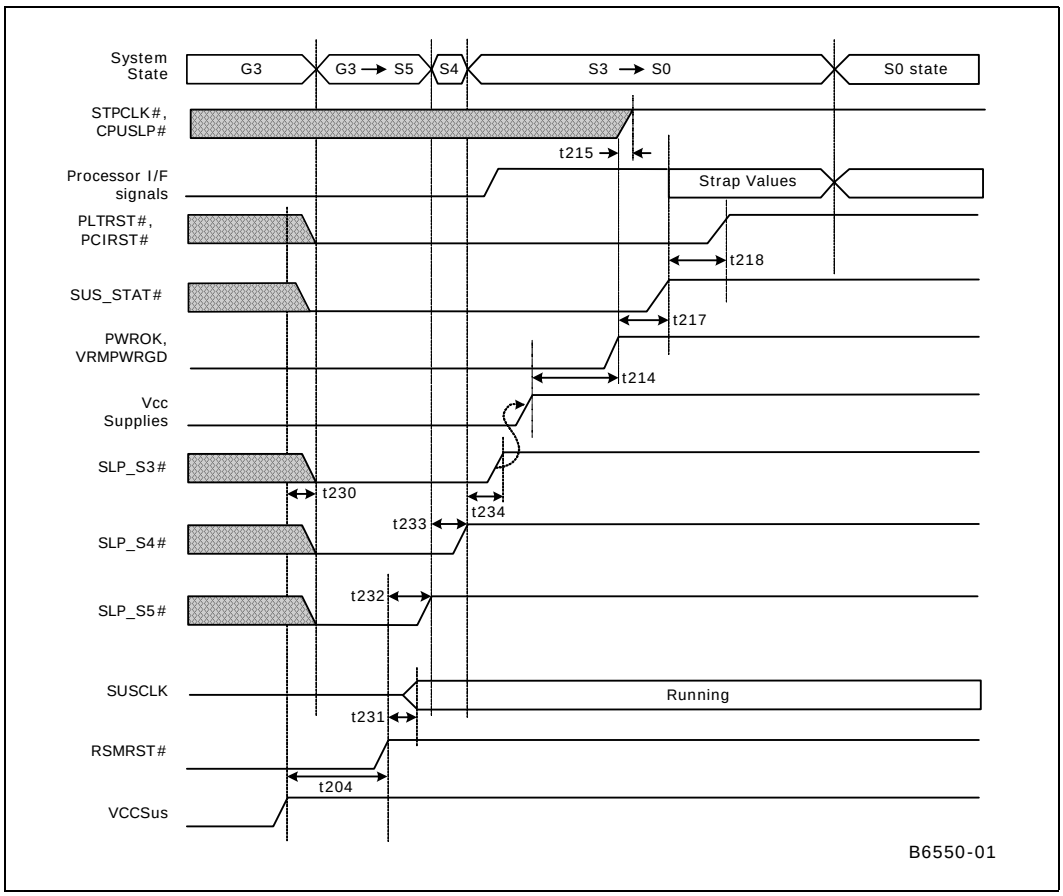


Figure 49-2. S0 to S1 to S0 Timing

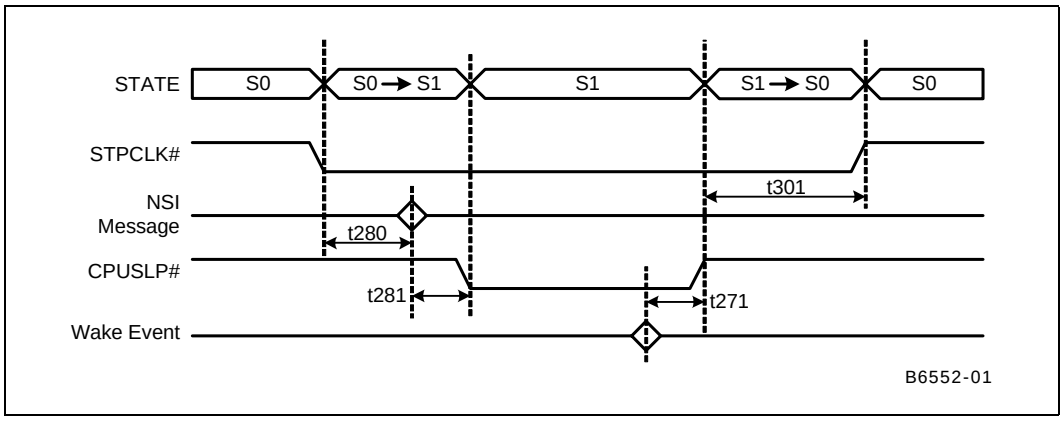
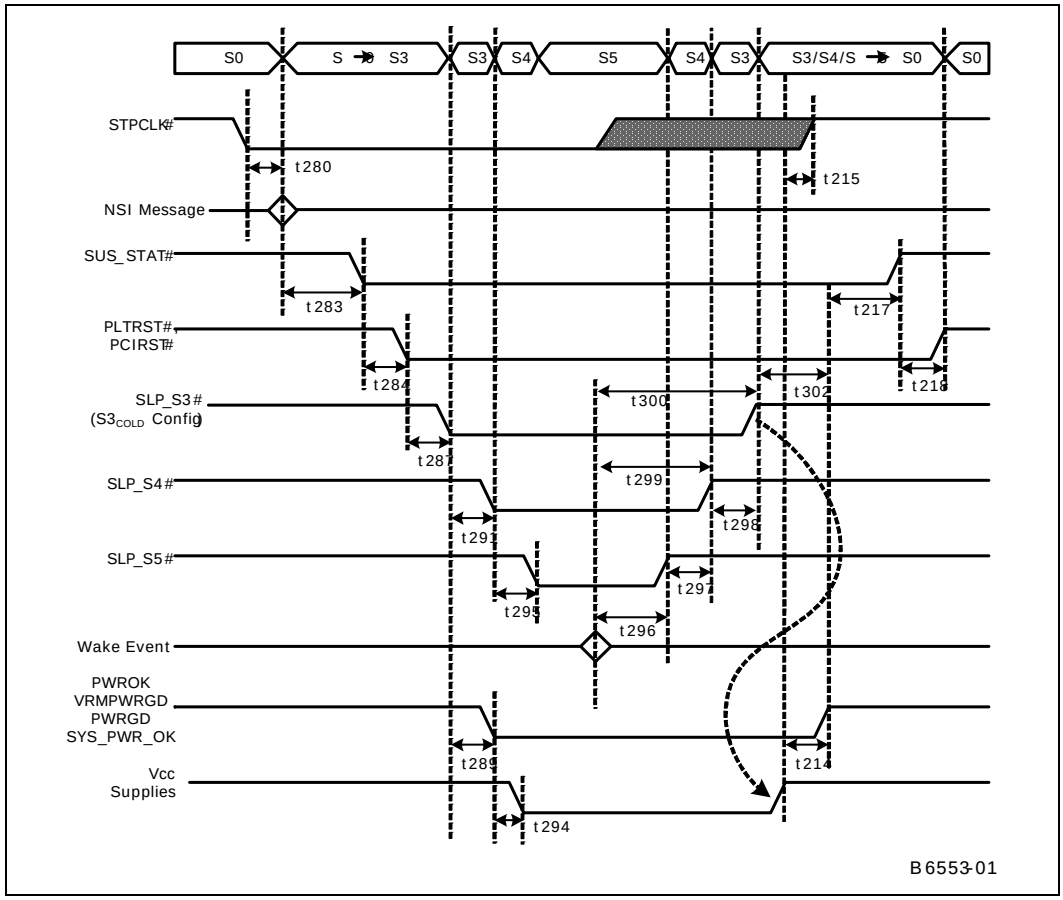


Figure 49-3. S0 to S5 to S0 Timings, $S3_{COLD}$



49.5.1.2.2 Power Management Miscellaneous Timing Diagrams

Refer to Section 6.1.3, "EP80579 Power Sequencing and Reset Sequence" for additional timing requirements related to power management signals.

49.5.2 DDR2

The EP80579 provides an integrated memory controller for direct-connection to a single channel of DDR2 with 400, 533, 640 and 800 MT/s unbuffered or registered memory devices, with a maximum support of up to two ranks.

The electrical specifications of the EP80579's DDR2 interface is compatible to the *JEDEC* Standard DDR2 SDRAM Specification, JESD79-2B, January 2005*. The AC & DC operating conditions are included in Chapter 5 of the JEDEC specification. Refer to the subsections that follow for the electrical characteristics of the EP80579 DDR2 interface.

49.5.2.1 DDR2 Signal List

For DDR2 pin descriptions, refer to Table 48-8, "DDR2 Interface Signals" on page 1739.

49.5.2.2 DDR2 DC Characteristics

Table 49-13. DDR2 DC Input Characteristics

Signal Group	Symbol	Parameter	Min	Nom	Max	Unit	Notes
DDR2 SSTL I/O	$V_{IL\ (DC)}$	Input Low Voltage (DC)	-	-	DDR_VREF - 150	mV	1, 2
	$V_{IH\ (DC)}$	Input High Voltage (DC)	DDR_VREF + 150	-	-	mV	1, 2
	I_{Leak}	Input Leakage Current (0<Vin<Vcc18)	-20	-	+20	µA	1

Notes:
1. Refer to the JESD-79-2 and SSTL-18 Specification for further details.
2. These input voltages apply only when the signals are inputs to the EP80579. When the signals are inputs to the SDRAM, the SDRAM input voltage specifications apply.

Table 49-14. DDR2 DC Output Characteristics (Sheet 1 of 2)

Signal Group	Symbol	Parameter	Min	Nom	Max	Unit	Notes
DDR2 Output	I_{OH}	Output High Current	-	-	-11.0	mA	1, 2, 3
	I_{OL}	Output Low Current	-	-	11.0	mA	1, 2, 3, 4
	V_{OL}	Output Low Voltage	0.0	-	DDR_VREF - 575	mV	1, 2
	V_{OH}	Output High Voltage	DDR_VREF + 575	-	-	mV	1, 2
	C_{Pin}	Pin Capacitance	2.0	-	2.5	pF	1, 5
	S_{out}	Output Slew Rate	2.2	-	3.2	V/ns	1, 6, 7

Notes:
1. Refer to the JESD-79-2 and SSTL-18 Specification for further details.
2. DDR2 DC parameters are specified with a 43 ohm test load to Vdd/2 and with up to 22 ohm resistive compensation (RCOMP). Guaranteed by design. These values are typical values seen for this process, but not measured during production testing
3. I_{OH}=0mA at V_{OH} of 0.99*V_{CC}; I_{OL}=0mA at V_{OL} of 0.01*V_{CC}
4. For open drain outputs, I_{OL}, MIN=3mA
5. Guaranteed by design.
6. Slew rate measured from vil(ac) to vih(ac)
7. The absolute value of the slew rate as measured from DC to DC is equal to or greater than the slew rate as measured from AC to AC. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-14. DDR2 DC Output Characteristics (Sheet 2 of 2)

Signal Group	Symbol	Parameter	Min	Nom	Max	Unit	Notes
	R_{out}	Output Impedance	20.0	-	40.0	Ohms	1
	DDR_VREF	DDR2 Reference Voltage	0.49 x Vcc18	0.50 x Vcc18	0.51 x Vcc18	V	1
VTTDDR	DDR_VTT	DDR2 Termination Voltage	DDR_VREF - 40	DDR_VREF	DDR_VREF + 40	mV	1
VCC18	DDR_VDD	DDR2 Supply Voltage	1.71	1.80	1.89	V	1

Notes:

1. Refer to the JESD-79-2 and SSTL-18 Specification for further details.
2. DDR2 DC parameters are specified with a 43 ohm test load to Vdd/2 and with up to 22 ohm resistive compensation (RCOMP). Guaranteed by design. These values are typical values seen for this process, but not measured during production testing
3. I_{OH}=0mA at V_{OH} of 0.99*V_{CC}; I_{OL}=0mA at V_{OL} of 0.01*V_{CC}
4. For open drain outputs, I_{OL}, MIN=3mA
5. Guaranteed by design.
6. Slew rate measured from vil(ac) to vih(ac)
7. The absolute value of the slew rate as measured from DC to DC is equal to or greater than the slew rate as measured from AC to AC. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

49.5.2.3 DDR2 AC Characteristics

Table 49-15. DDR2 Differential Input/Output AC Levels

Signal Group	Symbol	Parameter	Min	Max	Unit	Notes
DDR2 SSTL I/O	$V_{IL\ (AC)}$	Input Low Voltage (AC)	-	DDR_VREF – 200	mV	1, 2
	$V_{IH\ (AC)}$	Input High Voltage (AC)	DDR_VREF + 200	-	mV	1, 2
	V_{ID}	AC Differential Input Voltage	400	-	mV	
	V_{IX}	Differential cross point input voltage	DDR_VREF - 100	DDR_VREF + 100	mV	1, 3

Notes:
1. Refer to the JESD-79-2 and SSTL-18 Specification for further details.
2. These input voltages apply only when the signals are inputs to the EP80579. When the signals are inputs to the SDRAM, the SDRAM input voltage specifications apply.
3. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-16. DDR2-400 Interface AC Characteristics

Symbol	Parameter	Min	Max	Unit	Figures	Notes
System Memory Clock Timings						
T_{CK}	DDR_CK[5:0] Period	5	-	ns	49-9	
T_{CH}	DDR_CK[5:0] High Time	2.25	-	ns	49-11	
T_{CL}	DDR_CK[5:0] Low Time	2.25		ns	49-12	
tJIT	DDR_CK[5:0] Cycle to Cycle Jitter	-	225	ps		6
tSKEW_CK-CK	Skew between any two system memory differential clock pairs (DDR_CK[5:0]/DDR_CK[5:0]#)	-	$0.06T_{CK}$	ps	49-10	6
	DDR_CK[x] to DDR_CK[y]# (where x does not equal y)		$0.25T_{CK}$	ps	49-10	6
tSKEW_CK-DQS	Skew between any system memory clock pair and any system memory strobe	$-0.25T_{CK}$	$+0.25T_{CK}$	ps	49-13	6
System Memory Command and Control Signal Timings						
tCVB	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid Before DDR_CK Rising Edge	1.94	-	ns	49-8	
tCVA	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid After DDR_CK Rising Edge	1.94	-	ns	49-8	
System Memory Data and Strobe Signal Timings						
tDVB	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.828	-	ns	49-5	6
tDVA	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.828	-	ns	49-5	6
tDOPW	DDR_DQ[63:0], DDR_ECC[7:0] Output Valid Pulse Width	2.25	-	ns	-	3, 6
tSU_DQS	DDR_DQ and DDR_ECC Input Setup Time to DQS Crossing	–0.897	-	ns	49-4	1, 2, 6
tHD_DQS	DDR_DQ and DDR_ECC Input Hold Time After DQS Crossing	+1.60	-	ns	49-4	1, 6
tWPRE	DDR_DQS Write Preamble Duration	1.75 (nom)	-	ns	49-6	
tWPST	DDR_DQS Write Postamble Duration	2.00 (nom)	-	ns	49-7	

Notes:

1. Data to Strobe read setup and Data from strobe read hold minimum requirements specified are determined with the DQS Delay programmed for a 90 degree phase shift.
2. Refer to Figure 49-4. Negative minimum setup time at the EP80579 pins is correct – the DQS crossing at the EP80579 pins is expected to arrive before data becomes valid. Data is latched by a delayed copy of DQS (see Note 1), which nominally centers the strobe within the data valid window. Refer to the DDR2 Specification for further details.
3. TDOPW defines the minimum time a given data bit is guaranteed valid at the pin. Note that this is greater than the sum of minimum TDVA and TDVB, since strobe-to-data alignment uncertainty subtracts from those times.
4. AC timings are specified into a 25 ohm test load (for DQ and DQS) or 50 ohm test load (for other signals) terminated to Vdd/2
5. This specification applies for writes only; that is, when the IMCH is driving the strobes as well as the clocks. Refer to the JEDEC specification for an explanation of strobe to clock timing for DDR2 reads.
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-17. DDR2-533 Interface AC Characteristics

Symbol	Parameter	Min	Max	Unit	Figures	Notes
System Memory Clock Timings						
t_{CK}	DDR_CK[5:0] Period	3.75	-	ns	49-9	
t_{CH}	DDR_CK[5:0] High Time	1.70	-	ns	49-11	
t_{CL}	DDR_CK[5:0] Low Time	1.70		ns	49-12	
tJIT	DDR_CK[5:0] Cycle to Cycle Jitter	-	175	ps		6
tSKEW_CK-CK	Skew between any two system memory differential clock pairs (DDR_CK[5:0]/DDR_CK[5:0]#)	-	$0.06T_{CK}$	ps	49-10	6
	DDR_CK[x] to DDR_CK[y]# (where x does not equal y)		$0.25T_{CK}$	ps	49-10	6
tSKEW_CK-DQS	Skew between any system memory clock pair and any system memory strobe	$-0.25T_{CK}$	$+0.25T_{CK}$	ps	49-13	6
System Memory Command and Control Signal Timings						
tCVB	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid Before DDR_CK Rising Edge	1.379	-	ns	49-8	
tCVA	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid After DDR_CK Rising Edge	1.379	-	ns	49-8	
System Memory Data and Strobe Signal Timings						
tDVB	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	540.6	-	ps	49-5	6
tDVA	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	540.6	-	ps	49-5	6
tDOPW	DDR_DQ[63:0], DDR_ECC[7:0] Output Valid Pulse Width	1.70	-	ns	-	3, 6
tSU_DQS	DDR_DQ and DDR_ECC Input Setup Time to DQS Crossing	–0.594	-	ns	49-4	1, 2, 6
tHD_DQS	DDR_DQ and DDR_ECC Input Hold Time After DQS Crossing	+1.281	-	ns	49-4	1, 6
tWPRE	DDR_DQS Write Preamble Duration	1.31 (nom)	-	ns	49-6	
tWPST	DDR_DQS Write Postamble Duration	1.50 (nom)	-	ns	49-7	

Notes:

1. Data to Strobe read setup and Data from strobe read hold minimum requirements specified are determined with the DQS Delay programmed for a 90 degree phase shift.
2. Refer to Figure 49-5. Negative minimum setup time at the EP80579 pins is correct – the DQS crossing at the EP80579 pins is expected to arrive before data becomes valid. Data is latched by a delayed copy of DQS (see Note 1), which nominally centers the strobe within the data valid window. Refer to the DDR2 Specification for further details.
3. TDOPW defines the minimum time a given data bit is guaranteed valid at the pin. Note that this is greater than the sum of minimum TDVA and TDVB, since strobe-to-data alignment uncertainty subtracts from those times.
4. AC timings are specified into a 25 ohm test load (for DQ and DQS) or 50 ohm test load (for other signals) terminated to Vdd/2
5. This specification applies for writes only; that is, when the IMCH is driving the strobes as well as the clocks. Refer to the JEDEC specification for an explanation of strobe to clock timing for DDR2 reads.
6. Guaranteed by design.

Table 49-18. DDR2-667 Interface AC Characteristics

Symbol	Parameter	Min	Max	Unit	Figures	Notes
System Memory Clock Timings						
t_{CK}	DDR_CK[5:0] Period	3.125		ns	49-9	
t_{CH}	DDR_CK[5:0] High Time	1.562		ns	49-11	
t_{CL}	DDR_CK[5:0] Low Time	1.562		ns	49-12	
tJIT	DDR_CK[5:0] Cycle to Cycle Jitter		150	ps		6
tSKEW_CK-CK	Skew between any two system memory differential clock pairs (DDR_CK[5:0]/DDR_CK[5:0]#)	-	$0.06T_{CK}$	ps	49-10	6
	DDR_CK[x] to DDR_CK[y]# (where x does not equal y)		$0.25T_{CK}$	ps	49-10	6
tSKEW_CK-DQS	Skew between any system memory clock pair and any system memory strobe	$-0.25T_{CK}$	$+0.25T_{CK}$	ps	49-13	6
System Memory Command and Control Signal Timings						
tCVB	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid Before DDR_CK Rising Edge	1.04		ns	49-8	6
tCVA	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid After DDR_CK Rising Edge	1.04		ns	49-8	6
System Memory Data and Strobe Signal Timings						
tDVB	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.368		ns	49-5	6
tDVA	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.368		ns	49-5	6
tDOPW	DDR_DQ[63:0], DDR_ECC[7:0] Output Valid Pulse Width	1.35		ns	-	3, 6
tSU_DQS	DDR_DQ and DDR_ECC Input Setup Time to DQS Crossing	–0.413		ns	49-4	1, 2, 6
tHD_DQS	DDR_DQ and DDR_ECC Input Hold Time After DQS Crossing	+1.087		ns	49-4	1, 6
tWPRE	DDR_DQS Write Preamble Duration	1.05 (nom)		ns	49-6	6
tWPST	DDR_DQS Write Postamble Duration	1.20 (nom)		ns	49-7	6

Notes:

1. Data to Strobe read setup and Data from strobe read hold minimum requirements specified are determined with the DQS Delay programmed for a 90 degree phase shift.
2. Refer to Figure 49-5. Negative minimum setup time at the EP80579 pins is correct – the DQS crossing at the EP80579 pins is expected to arrive before data becomes valid. Data is latched by a delayed copy of DQS (see Note 1), which nominally centers the strobe within the data valid window. Refer to the DDR2 Specification for further details.
3. TDOPW defines the minimum time a given data bit is guaranteed valid at the pin. Note that this is greater than the sum of minimum TDVA and TDVB, since strobe-to-data alignment uncertainty subtracts from those times.
4. AC timings are specified into a 25 ohm test load (for DQ and DQS) or 50 ohm test load (for other signals) terminated to Vdd/2
5. This specification applies for writes only; that is, when the IMCH is driving the strobes as well as the clocks. Refer to the JEDEC specification for an explanation of strobe to clock timing for DDR2 reads.
6. Guaranteed by design.

Table 49-19. DDR2-800 Interface AC Characteristics

Symbol	Parameter	Min	Max	Unit	Figures	Notes
System Memory Clock Timings						
t_{CK}	DDR_CK[5:0] Period	2.5		ns	49-9	
t_{CH}	DDR_CK[5:0] High Time	1.120		ns	49-11	
t_{CL}	DDR_CK[5:0] Low Time	1.120		ns	49-12	
tJIT	DDR_CK[5:0] Cycle to Cycle Jitter		130	ps		6
tSKEW_CK-CK	Skew between any two system memory differential clock pairs (DDR_CK[5:0]/DDR_CK[5:0]#)	-	$0.06T_{CK}$	ps	49-10	6
	DDR_CK[x] to DDR_CK[y]# (where x does not equal y)		$0.25T_{CK}$	ps	49-10	6
tSKEW_CK-DQS	Skew between any system memory clock pair and any system memory strobe	$-0.25T_{CK}$	$+0.25T_{CK}$	ps	49-13	6
System Memory Command and Control Signal Timings						
tCVB	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid Before DDR_CK Rising Edge	0.819		ns	49-8	
tCVA	DDR_RAS#, DDR_CAS#, DDR_WE#, DDR_A[14:0], DDR_BA[2:0], DDR_CS[1:0]# Valid After DDR_CK Rising Edge	0.819		ns	49-8	
System Memory Data and Strobe Signal Timings						
tDVB	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.372		ns	49-5	6
tDVA	DDR_DQ[63:0], DDR_ECC[7:0], DDR_DM[8:0] Valid Before the corresponding DDR_DQS and DDR_DQS_L Crossing	0.372		ns	49-5	6
tDOPW	DDR_DQ[63:0], DDR_ECC[7:0] Output Valid Pulse Width	1.120		ns	-	3, 6
tSU_DQS	DDR_DQ and DDR_ECC Input Setup Time to DQS Crossing	–0.292		ns	49-4	1, 2, 6
tHD_DQS	DDR_DQ and DDR_ECC Input Hold Time After DQS Crossing	+0.956		ns	49-4	1, 6
tWPRE	DDR_DQS Write Preamble Duration	0.875 (nom)		ns	49-6	6
tWPST	DDR_DQS Write Postamble Duration	1.00 (nom)		ns	49-7	6

Notes:

1. Data to Strobe read setup and Data from strobe read hold minimum requirements specified are determined with the DQS Delay programmed for a 90 degree phase shift.
2. Refer to Figure 49-5. Negative minimum setup time at the EP80579 pins is correct – the DQS crossing at the EP80579 pins is expected to arrive before data becomes valid. Data is latched by a delayed copy of DQS (see Note 1), which nominally centers the strobe within the data valid window. Refer to the DDR2 Specification for further details.
3. TDOPW defines the minimum time a given data bit is guaranteed valid at the pin. Note that this is greater than the sum of minimum TDVA and TDVB, since strobe-to-data alignment uncertainty subtracts from those times.
4. AC timings are specified into a 25 ohm test load (for DQ and DQS) or 50 ohm test load (for other signals) terminated to Vdd/2
5. This specification applies for writes only; that is, when the IMCH is driving the strobes as well as the clocks. Refer to the JEDEC specification for an explanation of strobe to clock timing for DDR2 reads.
6. Guaranteed by design.

49.5.2.3.1 DDR2 AC Timing Diagrams

Figure 49-4. DQ and CB (ECC) Setup / Hold Relationship to / from DQS (Read Operation)

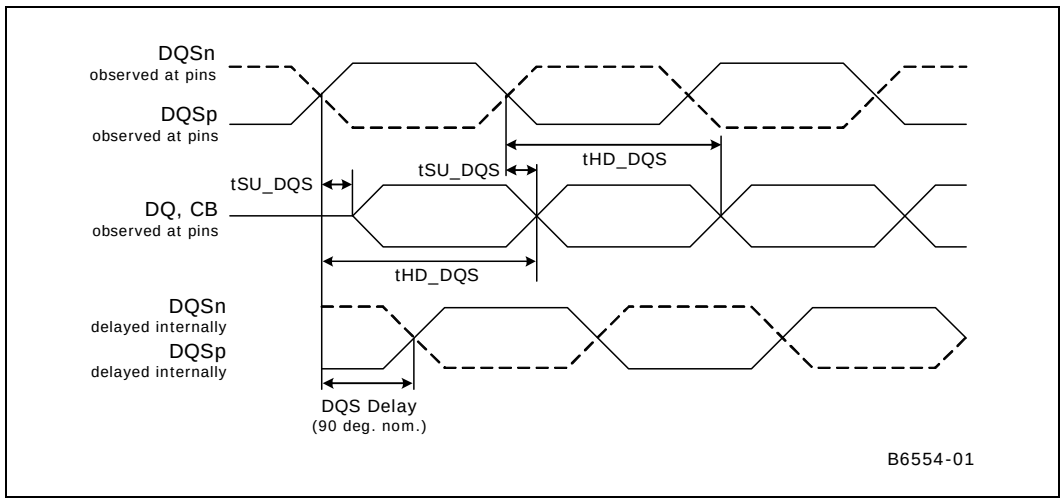


Figure 49-5. DQ and CB (ECC) Valid Before and After DQS (Write Operation)

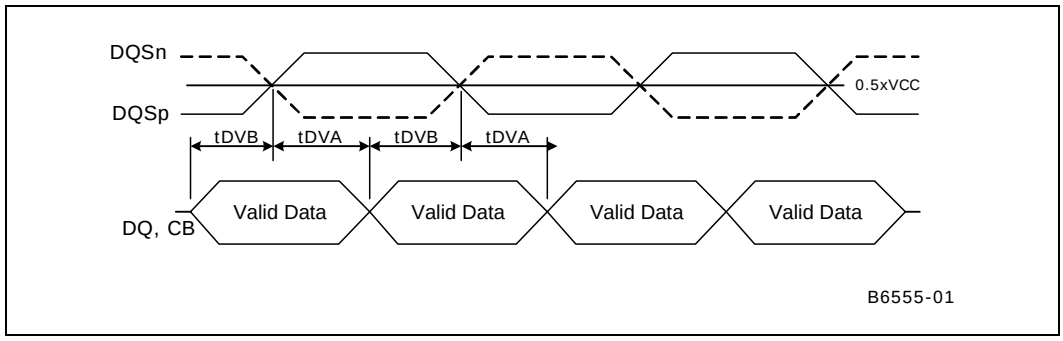


Figure 49-6. Write Preamble Duration

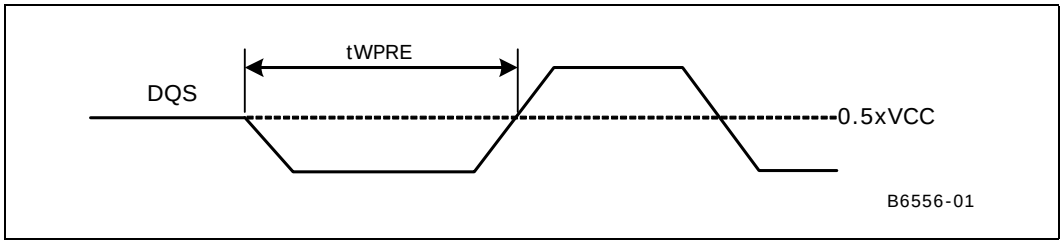


Figure 49-7. Write Postamble Duration

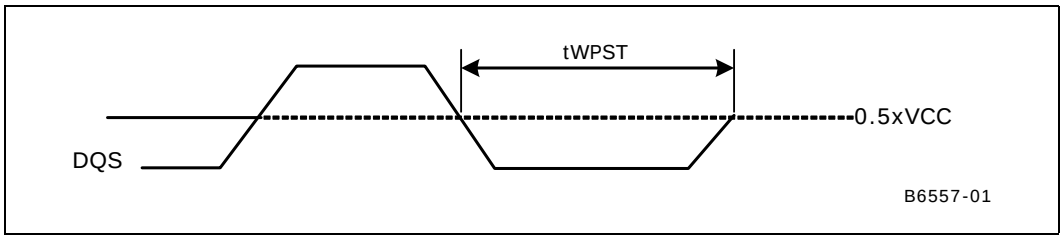


Figure 49-8. Control Signals Valid before and after DDR_CK Rising Edge

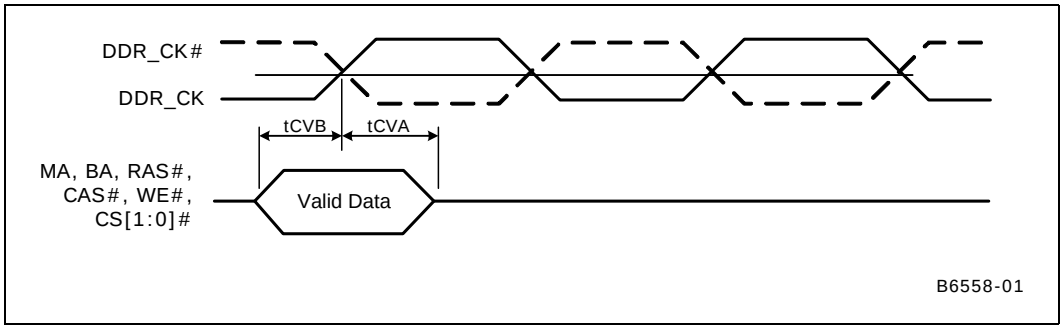


Figure 49-9. Clock Cycle Time

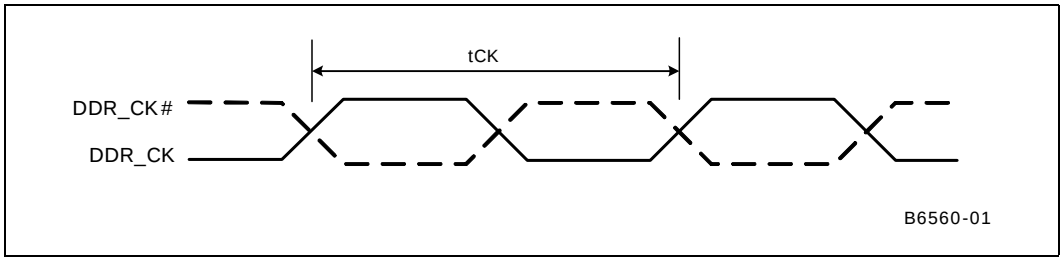
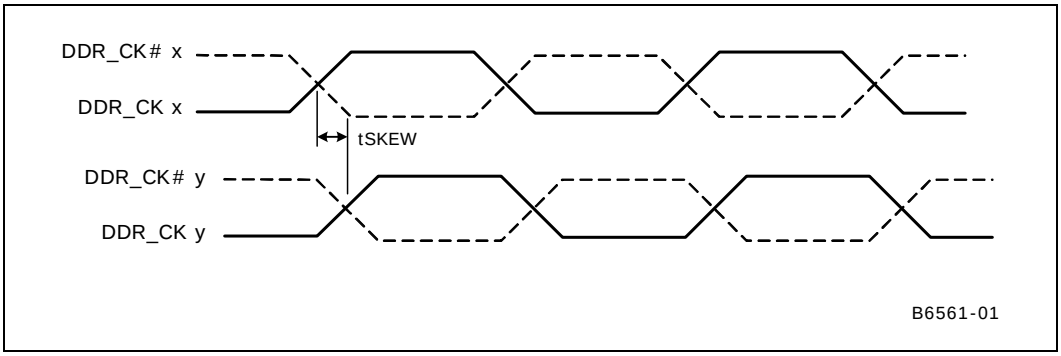


Figure 49-10.Skew Between any System Memory Differential Clock Pair (DDR_CK/ DDR_CK#)



Note: *x* represents one differential clock pair, and y represents another differential clock pair.

Figure 49-11.DDR2 Command Clock High Time

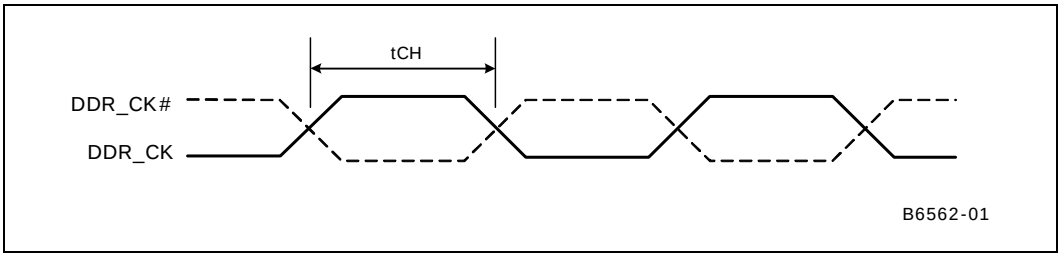


Figure 49-12.DDR2 Command Clock Low Time

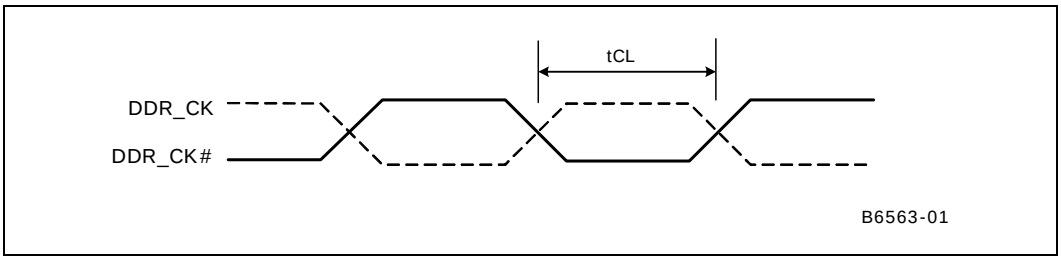
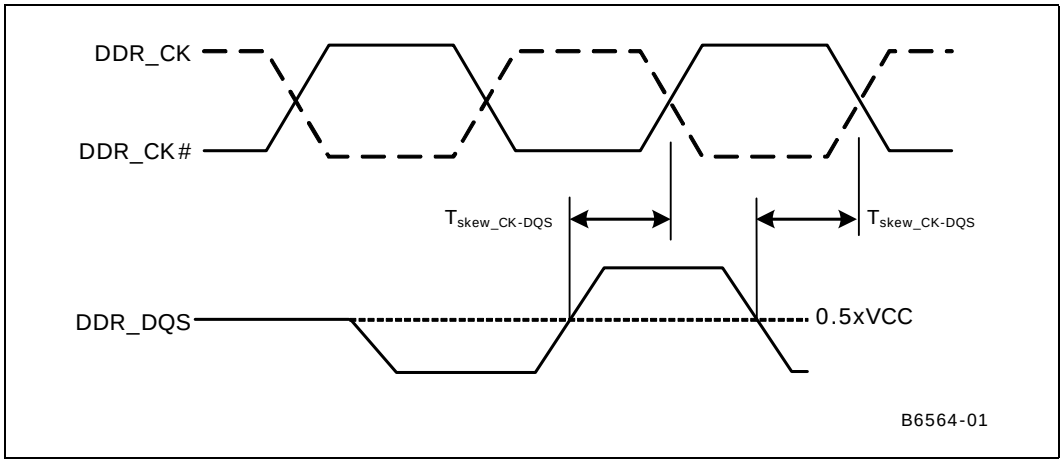


Figure 49-13.DDR2 Command Clock to DQS skew



49.5.3 PCI Express*

This section describes the electrical characteristics of the PCI Express* interface.

The PCI Express* interface signals are driven by transceivers designed specifically for high-speed serial communication. All PCI Express* signals are fully differential and operate in a current mode, rather than a voltage mode. These interfaces support A/C coupling to facilitate communication across independent power supply domains and signal at a rate well above the flight time of the interface. The nominal supply voltage for the PCI Express* interfaces is 1.2V, although receivers must tolerate up to 1.6V peak-to-peak across differential pairs as required by the *PCI Express* Interface Specification*.

This section describes the electrical characteristics of the PCI Express* interface.

49.5.3.1 PCI Express* Signal List

For PCI Express* pin description refer to Table 48-9, “PCI Express Interface Signals” on page 1741.

49.5.3.2 PCI Express* Differential Transmitter and Receiver Specifications

This electrical specifications of the PCI Express* Differential Transmitter and Receiver in the EP80579 are compliant with the *PCI Express* Base Specification, Rev. 1.1* (http://www.pcisig.com). Refer to the specification subsections that follow for more information:

- Section 4.3 Electrical Sub-Block of *PCI Express* Base Specification, Rev. 1.1*:
  - — Section 4.3.1 Electrical Sub-Block Requirements
  - — Section 4.3.2 Electrical Signal Specifications
  - — Section 4.3.3 Differential Transmitter (Tx) Output Specifications
  - — Section 4.3.4 Differential Receiver (Rx) Input Specifications
  - — Section 4.3.3.1 Tx Compliance Eye Diagrams
  - — Section 4.3.3.1 Rx Compliance Eye Diagrams

Table 49-20. PCI Express* Differential Receiver (RX) Specifications

Symbol	Parameter	Min	Nom	Max	Unit	Figures	Notes
$Z_{RX-DIFF-DC}$	Rx DC Differential Input Impedance	80	100	120	Ohms	-	1
Z_{RX-DC}	Rx DC Input Common Mode Impedance	40	50	60	Ohms	-	1, 2
$Z_{RX-HIGH-IMP-DC}$	Rx Powered Down DC Input Common Mode Impedance	200k	-	-	Ohms	-	3
UI	Unit Interval	399.88	400	400.12	ps	-	5
$V_{RX-DIFFp-p}$	Differential Peak-to-Peak Input Voltage (at spec load)	0.175	-	1.2	V	49-15	6, 4
T_{RX-EYE}	Minimum Receiver Eye Width	0.40	-	-	UI	49-15	6, 7
$T_{RX-EYE-MEDIAN-to-MAX-JITTER}$	Maximum time between the jitter median and maximum deviation from the median	-	-	0.3	UI	-	6, 7
$V_{RX-CM-ACp}$	AC Peak Common Mode Input Voltage	-	-	150	mV	-	6
$RL_{RX-DIFF}$	Differential Return Loss	15	-		dB	-	8
RL_{RX-CM}	Common Mode Return Loss	6	-		dB	-	8
$V_{RX-IDLE-DET-DIFFp-p}$	Electrical Idle Detect Threshold	65	-	175	mV	-	
$T_{RX-IDLE-DET-DIFF-ENTERTIME}$	Unexpected Electrical Idle Enter Detect Threshold Integration Time	-	-	10	ms	-	
$L_{RX-SKEW}$	Total Skew	-		20	ns	-	

Notes:

1. Specified at the measurement point and measured over any 250 consecutive UIs. The test load in Figure 49-14 (not the EP80579 itself) must be used as the Rx device when taking measurements (also refer to the Receiver Compliance Eye Diagram as shown in Figure 49-15). If the clocks to the Rx and Tx are not derived from the same clock chip the Tx UI must be used as a reference for the eye diagram.
2. Impedance during all LTSSM states. When transitioning from a Fundamental Reset to Detect (the initial state of the LTSSM) there is a 5ms transition time before Receiver termination values must be met on all un-configured Lanes of a Port.
3. The Rx DC Common Mode Impedance that exists when no power is present or Fundamental Reset is asserted. This helps ensure that the Receiver Detect circuit does not falsely assume a Receiver is powered on when it is not. This term must be measured at 300mV above the Rx ground.
4. PCI-Express mVdiff p-p = PEA0_Xp[x] - PEA0_Xn[x]
5. No test load is necessarily associated with this value.
6. Specified at the measurement point and measured over any 250 consecutive UIs. The test load in Figure 49-14 and measured over any 250 consecutive Unit Intervals. Also refer to the Receiver Compliance Eye Diagram as shown in Figure 49-15. That is, the receiver device must be replaced with 50 Ω terminations to ground on each half of the signal pair for the purpose of measuring this parameter. If the clocks to the Rx and Tx are not derived from the same clock chip the Tx UI must be used as a reference for the eye diagram.
7. A TRX-EYE = 0.40 UI provides for a total sum of 0.60 UI deterministic and random jitter budget for the transmitter and interconnect collected any 250 consecutive UIs. The TRX-EYE-MEDIAN-to-MAX-JITTER specification ensures a jitter distribution in which the median and the maximum deviation from the median is less than half of the total.6 UI jitter budget collected over any 250 consecutive Tx UIs. It must be noted that the median is not the same as the mean. The jitter median describes the point in time where the number of jitter points on either side is approximately equal as opposed to the averaged time value. If the clocks to the Rx and Tx are not derived from the same clock chip, the appropriate average Tx UI must be used as the reference for the eye diagram.
8. The receiver input impedance shall result in a differential return loss greater than or equal to 15 dB and a common mode return loss greater than or equal to 6 dB over a frequency range of 50 MHz to 1.25 GHz. This input impedance requirement applies to all valid input levels. The reference impedance for return loss measurements for is 50 Ω to ground for both the D+ and D– line (for example, as measured by a Vector Network Analyzer with 50 Ω probes – see Figure 49-15). Note that the series capacitors CTX is optional for the return loss measurement.

Table 49-21. PCI Express* Differential Transmitter (TX) Specifications (Sheet 1 of 2)

Symbol	Parameter	Min	Nom	Max	Unit	Figures	Notes
$V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$	Tx Absolute Delta of DC Common Mode Voltage During L0 and Electrical Idle	0	-	100	mV	-	1
$V_{TX-CM-DC-LINE-DELTA}$	Tx Absolute Delta of DC Common Mode Voltage between D+ and D–	0	-	25	mV	-	1
$V_{TX-DC-CM}$	Tx DC Common Mode Voltage	0	-	3.6	V	-	2
$I_{TX-SHORT}$	Tx Short Circuit Current Limit		-	90	mA	-	4
$Z_{TX-DIFF-DC}$	Tx DC Differential Impedance	80	100	120	Ohms	-	
Z_{TX-DC}	Tx DC impedance	40	-	-	Ohms	-	
$L_{TX-SKEW}$	Tx Lane-to-Lane Output Skew	-	-	500 ps + 2UI		-	3
C_{TX}	Tx AC Coupling Capacitor	75	-	200	nF	-	6
V_{O7}	Tx Output Voltage	800	-	1200	mVdiff p-p	-	5
UI	Unit Interval	399.88	400	400.12	ps	-	7
$V_{TX-DIFFp-p}$	Differential Peak-to-Peak Output Voltage	0.800	-	1.2	V	49-16	8
$V_{TX-DE-RATIO}$	De-emphasized Differential Output Voltage Ratio	–3.0	–3.5	–4.0	dB	49-16	8
T_{TX-EYE}	Minimum Tx Eye Width	0.70	-		UI	49-16	8, 9
$T_{TX-EYE-MEDIAN-to-MAX-JITTER}$	Maximum time between the jitter median and maximum deviation from the median	-	-	0.15	UI	-	8, 9
$T_{TX-RISE}$, $T_{TX-FALL}$	D+/D– Tx Output Rise/Fall time	0.125	-	-	UI	-	8, 11
$V_{TX-CM-ACp}$	AC Peak Common Mode Output Voltage		-	20	mV	-	8

Notes:

1. Specified at the measurement point into a timing and voltage compliance test load as shown in Figure 49-14 and measured over any 250 consecutive Unit Intervals. Also refer to the Transmitter Compliance Eye Diagram as shown in Figure 49-15.
2. The allowed DC Common Mode voltage under any conditions. Refer to Section 4.3.1.8 in the *PCI-Express* Specification* for further details.
3. Static skew between any two Transmitter Lanes within a single link
4. The allowed current when any output is shorted to ground.
5. PCI-Express mVdiff p-p = PEx_Xp[x] - PEx_Xn[x]
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.
7. No test load is necessarily associated with this value.
8. Specified at the measurement point into a timing and voltage compliance test load as shown in Figure 49-14 and measured over any 250 consecutive Unit Intervals. Also refer to the Transmitter Compliance Eye Diagram as shown in Figure 49-16.
9. A TTX-EYE = 0.70 UI provides for a total sum of deterministic and random jitter budget of TTX-JITTER-MAX = 0.30 UI for the transmitter collected over any 250 consecutive Tx UIs. The TTX-EYE-MEDIAN-to-MAX-JITTER specification ensures a jitter distribution in which the median and the maximum deviation from the median is less than half of the total Tx jitter budget collected over any 250 consecutive Tx UIs. Note that the median is not the same as the mean. The jitter median describes the point in time where the number of jitter points on either side is approximately equal, as opposed to the average value.
10. The transmitter input impedance shall result in a differential return loss greater than or equal to 12 dB and a common mode return loss greater than 6 dB over a frequency range of 50 MHz to 1.25 GHz. This input impedance requirement applies to all valid input levels. The reference impedance for return less measurements is 50 Ω to ground for both the D+ and D– line (for example, as measured by a Vector Network Analyzer with 50 Ω probes – see Figure 49-16). Note that the series capacitors CTX is optional for the return loss measurement.
11. Measured between 20–80% at Transmitter package pins into a test load as shown in Figure 49-16 for both VTX-D+ and VTX-D–.
12. Refer to Section 4.3.1.8 in the *PCI-Express* Specification* for further details.

Table 49-21. PCI Express* Differential Transmitter (TX) Specifications (Sheet 2 of 2)

Symbol	Parameter	Min	Nom	Max	Unit	Figures	Notes
$V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$	Absolute Delta of DC Common Mode Voltage During L0 and Electrical Idle	0		100	mV	-	8
$V_{TX-CM-DC-LINE-DELTA}$	Absolute Delta of DC Common Mode Voltage between D+ and D–	0		20	mV		8
$V_{TX-IDLE-DIFFp}$	Electrical Idle Differential Peak Output Voltage	0		20	mV		8
$V_{TX-RCV-DETECT}$	The amount of voltage change allowed during Receiver Detection			600	mV		12
$T_{TX\_IDLE-MIN}$	Minimum time spent in Electrical Idle	50			UI		
$T_{TX-IDLE-SET-TO-IDLE}$	Maximum time to transition to a valid Electrical Idle after sending an Electrical Idle ordered set			20	UI		
$T_{TX-IDLE-TO-DIFF-DATA}$	Maximum time to transition to valid Tx Specification after leaving an Electrical Idle condition			20	UI		
$T_{TX-IDLE-RCV-DETECT-MAX}$	Maximum time spent in electrical Idle before initiating a receiver detect sequence			100	ms		12
$RL_{TX-DIFF}$	Differential Return Loss	10			dB		10
RL_{TX-CM}	Common Mode Return Loss	6			dB		10
$T_{crosslink}$	Crosslink Random Timeout	0		1	ms		

Notes:

1. Specified at the measurement point into a timing and voltage compliance test load as shown in Figure 49-14 and measured over any 250 consecutive Unit Intervals. Also refer to the Transmitter Compliance Eye Diagram as shown in Figure 49-15.
2. The allowed DC Common Mode voltage under any conditions. Refer to Section 4.3.1.8 in the *PCI-Express* Specification* for further details.
3. Static skew between any two Transmitter Lanes within a single link
4. The allowed current when any output is shorted to ground.
5. PCI-Express mVdiff p-p = PEx_Xp[x] - PEx_Xn[x]
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.
7. No test load is necessarily associated with this value.
8. Specified at the measurement point into a timing and voltage compliance test load as shown in Figure 49-14 and measured over any 250 consecutive Unit Intervals. Also refer to the Transmitter Compliance Eye Diagram as shown in Figure 49-16.
9. A TTX-EYE = 0.70 UI provides for a total sum of deterministic and random jitter budget of TTX-JITTER-MAX = 0.30 UI for the transmitter collected over any 250 consecutive Tx UIs. The TTX-EYE-MEDIAN-to-MAX-JITTER specification ensures a jitter distribution in which the median and the maximum deviation from the median is less than half of the total Tx jitter budget collected over any 250 consecutive Tx UIs. Note that the median is not the same as the mean. The jitter median describes the point in time where the number of jitter points on either side is approximately equal, as opposed to the average value.
10. The transmitter input impedance shall result in a differential return loss greater than or equal to 12 dB and a common mode return loss greater than 6 dB over a frequency range of 50 MHz to 1.25 GHz. This input impedance requirement applies to all valid input levels. The reference impedance for return less measurements is 50 Ω to ground for both the D+ and D– line (for example, as measured by a Vector Network Analyzer with 50 Ω probes – see Figure 49-16). Note that the series capacitors CTX is optional for the return loss measurement.
11. Measured between 20–80% at Transmitter package pins into a test load as shown in Figure 49-16 for both VTX-D+ and VTX-D–.
12. Refer to Section 4.3.1.8 in the *PCI-Express* Specification* for further details.

49.5.3.3 PCI Express* Clock Specifications

Table 49-22. PCI Express* Clock DC Specifications

Symbol	Parameter	Min	Nom	Max	Unit	Notes
V_{IL}	Input Low Voltage	–0.150	0	0.150	V	1
V_{IH}	Input High Voltage	0.660	0.710	0.850	V	1
$V_{CROSS(abs)}$	Absolute Crossing Point	0.250		0.550	V	1, 2, 6
$V_{CROSS(rel)}$	Relative Crossing Point	0.250 + 0.5 x (V_{Havg} – 0.710)		0.550 + 0.5 x (V_{Havg} – 0.710)	V	1, 3, 6, 7
ΔV_{CROSS}	Range of Crossing Points			0.140	V	1
V_{RBM}	Ringback Margin	0.200			V	1, 4
V_{TR}	Threshold Region	V_{CROSS} – 0.100		V_{CROSS} + 0.100	V	1, 5

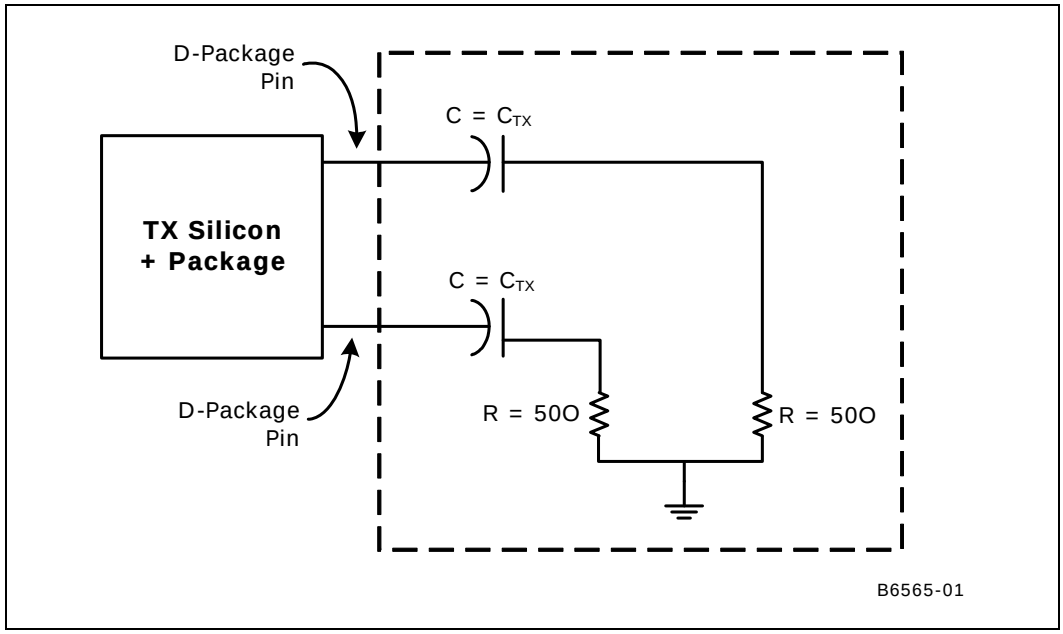
Notes:
1. Refer to Figure 49-17 and Figure 49-18.
2. Crossing voltage is defined as the instantaneous voltage when the rising edge of PEA_CLKp is equal to the falling edge of PEA_CLKn.
3. VHavg is the statistical average of the VH measured by the oscilloscope.
4. Ringback Margin is defined as the absolute voltage difference between the maximum Rising Edge Ringback and the maximum Falling Edge Ringback.
5. Threshold Region is defined as a region entered around the crossing point voltage in which the differential receiver switches. It includes input threshold hysteresis.
6. The crossing point must meet the absolute and relative crossing point specifications simultaneously.
7. VHavg can be measured directly using “Vtop” on Agilent* scopes and “High” on Tektronix* scopes.

Table 49-23. PCI Express* Clock Timings

Symbol	Parameter	Min	Typical	Max	Units	Figure	Notes
	PEA_CLK Frequency		100		MHz		
	Clock Tolerance	300		300	ppm		
Tp	PEA_CLK Period	9.872			ns	49-17	
Tccjitter	Cycle to Cycle Jitter			125	ps		
	Duty Cycle	45		55	%		
T5	PEA_CLK Rise Time	175		700	ps	49-17	
ΔTrise	PEA_CLK Rise Time variation			125	ps		
T6	PEA_CLK Fall Time	175		700	ps	49-17	
ΔTfall	PEA_CLK Fall Time variation			125	ps		
	PEA_CLK Slew Rate	0.5		1.6	V/ns		1, 2
	Rise/Fall Matching			20	%		

Notes:
1. Rise/fall times and slew rates are measured single ended between 245 mV and 455 mV of the clock swing.
2. Slew rate specifications apply to both rising and falling edges.

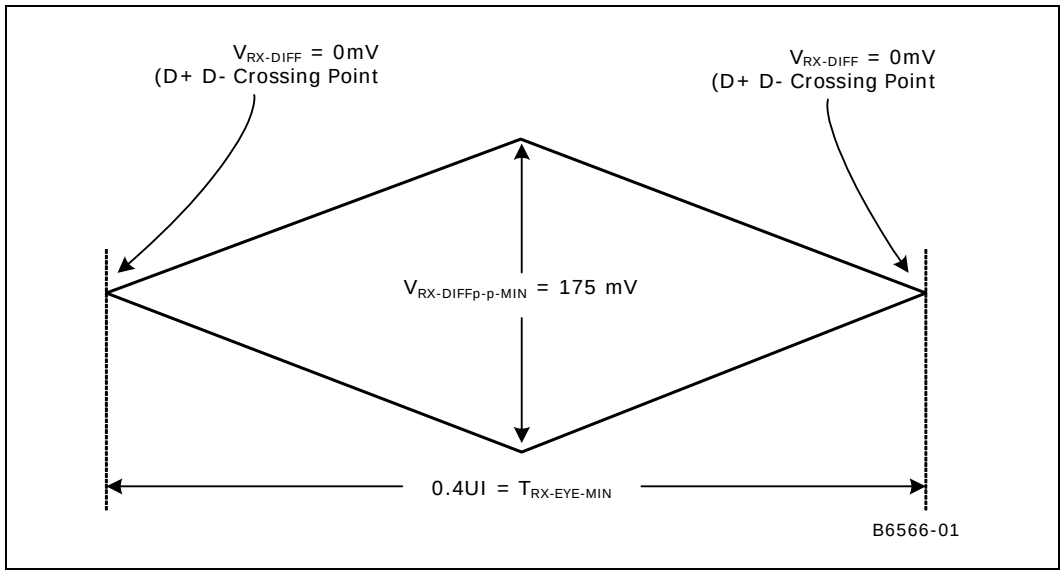
Figure 49-14.PCI Express* Transmitter Test Load



49.5.3.3.1 PCI Express* Receiver Compliance Eye Diagram

Figure 49-15 shows the PCI Express* Rx compliance eye diagram. For more details related to the Rx compliance eye diagrams refer to Section 4.3.4 of the *PCI Express* Base Specification, Rev. 1.1.*

Figure 49-15.PCI Express* Receiver Compliance Eye Diagram



49.5.3.3.2 PCI Express* Transmitter Compliance Eye Diagram

Figure 49-16 shows the PCI Express* Tx compliance eye diagram. For more details related to the transmit compliance eye diagrams refer to Section 4.3.3.1 of the *PCI Express* Base Specification, Rev. 1.1.*

Figure 49-16.PCI Express* Transmitter Compliance Eye Diagram

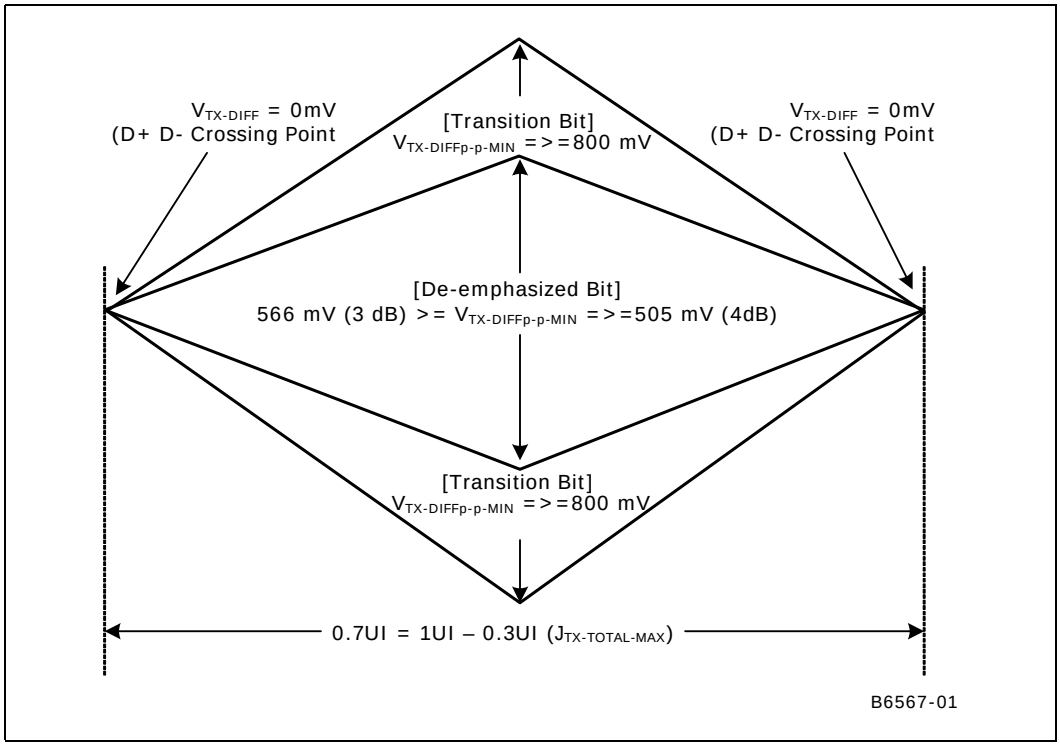
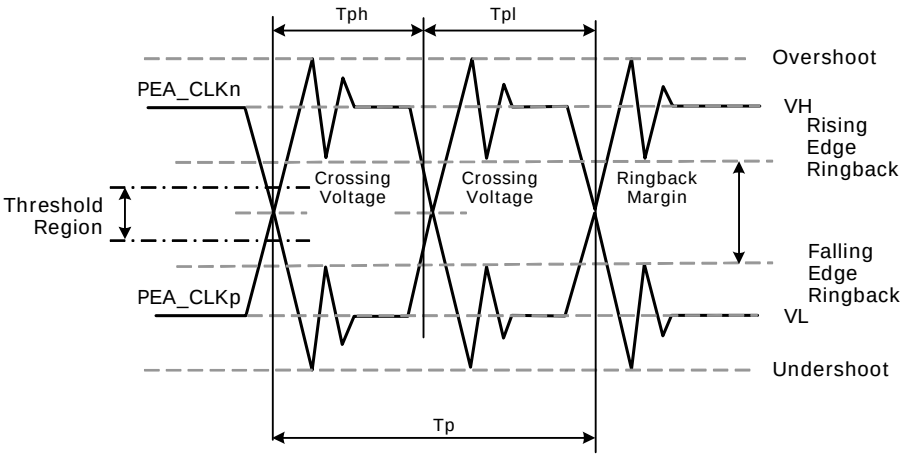


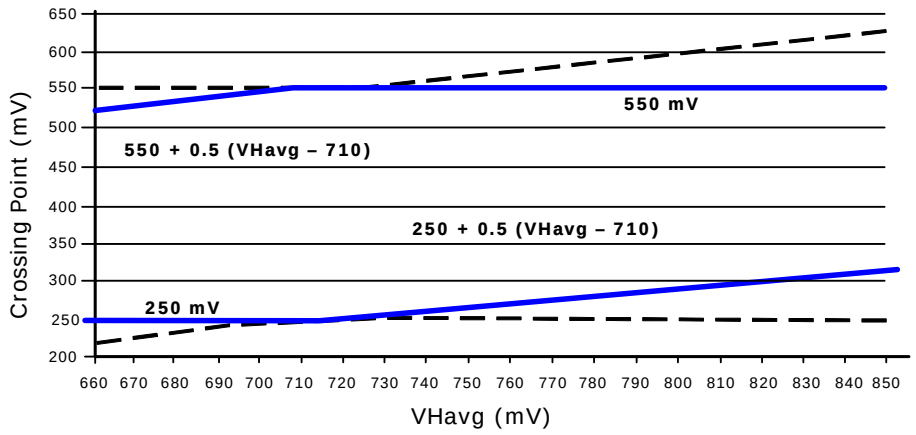
Figure 49-17.Differential Clock Waveform



Tp = T1: PEA_CLK period
T2: PEA_CLK period stability (not shown)
Tph = T3: PEA_CLK pulse high time
Tpl = T4: PEA_CLK pulse low time
T5: PEA_CLK rise time through the threshold region
T6: PEA_CLK fall time through the threshold region

B6568-01

Figure 49-18.Differential Clock Cross-Point Specification



B6569-01

49.5.4 Serial ATA (SATA)

The EP80579 has an integrated SATA host controller that supports independent DMA operation on two ports and supports data transfer rates of up to 3.0 Gb/s (300 MB/s). This technology consists of a receiver and transmitter.

The EP80579 SATA ports support Gen1 Electrical Specifications for “Short” Backplane and External Desktop Applications running at 1.5Gbp/s. These are defined on page 12 of the *Serial ATA II: Electrical Specification Revision 1.0 26-May 2004*. The EP80579 silicon and platform supports SATA hot plug. Use of an interlock switch is optional. While SATA controllers commonly provide hot plug functionality, SATA cable assemblies are not standardized, especially the SATA power cable. To support SATA hot plug, any EP80579 platform needs to support a SATA 15-pin power connector interface with a hot plug compatible SATA HDD.

The EP80579 supports the *Serial ATA Specification, Revision 1.0a*. The EP80579 also supports several optional sections of the *Serial ATA II: Extensions to Serial ATA 1.0 Specification, Revision 1.0* (AHCI support is required for some elements).

This section describes the electrical characteristics of the SATA interface.

49.5.4.1 SATA Signal List

For SATA pin descriptions, refer to Table 48-17, “Serial ATA Interface Signals” on page 1753.

49.5.4.2 SATA DC Characteristics

Table 49-24. SATA DC Input Characteristics (SATA_RX[P,N])

Symbol	Parameter	Conditions	Gen 1 m	Gen 2 m	Units	Notes
V_{IMIN10}	Minimum Input Voltage		240	240	mVdiff p-p	1
V_{IMAX10}	Maximum Input Voltage		600	750	mVdiff p-p	1

Notes:
1. SATA Vdiff, tx (VIMAX/MIN10) is measured at the SATA connector on the transmit side (generally, the motherboard connector), where SATA mVdiff p-p = |SATA[x]TXP/RXP - SATA[x]TXN/RXN|

Table 49-25. SATA DC Input Characteristics (GPIO and SATALED#)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.4.3 SATA DC Output Characteristics

Table 49-26. SATA DC Output Characteristics (SATA_TX(P,N))

Symbol	Parameter	Conditions	Gen 1 m	Gen 2 m	Units	Note
V_{OMIN8}	Minimum Output Voltage		400	400	mVdiff p-p	1
V_{OMAX8}	Maximum Output Voltage		600	700	mVdiff p-p	1

Notes:
1. SATA Vdiff, Tx (VOMAX/MIN8) is measured at the SATA connector on the transmit side closest to the IICH (generally, the motherboard connector), where SATA mV diff p-p = |SATAxTXP - SATAxTXN

Table 49-27. SATA DC Output Characteristics (SATALED#)

Symbol	Parameter	Conditions	Min	Typical	Max	Units
V_{OH}	Output voltage high	I_{out}=-6mA	2.4	-		V
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V
I_{OH}	Output current at high voltage	V_{OH}=2.4	-	-	-6	mA
I_{OL}	Output current at low voltage	V_{OL}=0.8	2	-	-	mA

49.5.4.3.1 SATA_CLK

The 100MHz Serial ATA Reference Clock (SATACLKP, SATACLKN) is the main clock for the SATA interface. The reference clock is used by the SATA PLL to generate the clocks of the SATA interface and is implemented on the system as a ground-terminated, low-voltage differential signal pair driven by the system clock chip.

Table 49-28. SATA DC Clock Specifications (SATA_CLKN, SATA_CLKP)

Symbol	Parameter	Conditions	Min	Typical	Max	Units
VIL12	Input Low Voltage		-0.150		0.150	V
VIH12	Input High Voltage		0.660		0.850	V
Vcross(abs)	Absolute Crossing Point		0.250		0.550	V

49.5.4.4 SATA LED

The SATALED# output is driven whenever the BSY bit is set in any SATA port. The SATALED# is an active-low open-collector output. When SATALED# is low, the LED is active. When SATALED# is high, the LED is inactive.

49.5.4.5 SATA AC Characteristics

Table 49-29. SATA Clock (SATA_CLKp, SATA_CLKn)

Parameter	Min	Typical	Max	Units
Period	9.997	-	10.0533	ns
Rise time	175	-	700	ps
Fall time	175	-	700	ps

Table 49-30. SATA Interface Timings

Symbol	Parameter	Min	Typical	Max	Units	Notes
UI	Gen1 Operating Data Period	666.43	666.6667	670.12	ps	5
UI	Gen2 Operating Data Period (3 Gbps)	333.21	333.3333	335.06	ps	5
	Rise Time (Gen1)	0.10	-	0.41	UI	1, 5
	Fall Time (Gen1)	0.10	-	0.41	UI	2, 5
	Rise Time (Gen2)	0.20	0.3	0.41	UI	1, 5
	Fall Time (Gen2)	0.20	0.3	0.41	UI	2, 5
	Tx differential skew	-	-	20	ps	
	COMRESET	310.4	320	329.6	ns	3
	COMWAKE transmit spacing	103.5	106.7	109.9	ns	3
	OOB Operating Data period	646.67	-	686.67	ns	4

Notes:
1. 20% - 80% at transmitter
2. 80% - 20% at transmitter
3. As measured from 100 mV differential cross points of last and first edges of burst.
4. Operating data period during Out-Of-Band burst transmissions
5. Unit Interval (UI). Equal to the time required to transmit one bit.

49.5.5 Universal Serial Bus (USB)

The EP80579 has (2) Universal Serial Bus 2.0 interface that can be configured as master, but not as a slave interface.

The USB interface supports both USB 1.1 and USB 2.0 signaling. USB1.1 has a low-speed mode (1.5 Mbps) and a full-speed mode (12 Mbps). USB 2.0 works at high speed LVDS mode with a data rate of 480 Mbps. The driver operates in voltage mode for USB 1.1 and current mode for USB 2.0. The receiver can detect single-ended/ differential-ended classical signals as well as high speed LVDS signals.

This section describes the electrical characteristics of the USB interface.

49.5.5.1 USB Signal List

For USB pin description refer to Table 48-18, “USB Interface Signals” on page 1755.

49.5.5.2 USB DC Characteristics

49.5.5.2.1 USB Transreceiver DC Characteristics

The DC electrical specifications of the USB Transreceivers in the EP80579 (USBp0, USBn0, USBp1, USBn1) are compliant with the USB v2.0 specification from April 27, 2000 (refer to Section 7.3.2 Bus Timing/Electrical Characteristics for more information).

External resistors are not required on USBp0, USBn0, USBp1 and USBn1. The chip integrates the 15 KΩ pull-down and provides an output driver impedance of 45 Ohm, which requires no external series resistor.

49.5.5.2.2 USB LVTTL Inputs DC Characteristics

The DC electrical specifications for the LVTTL inputs are described in Table 49-31.

Table 49-31. USB Overcurrent Indicators DC Input (OC[1:0])

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	Vcc3_3 +0.3V	V	-
V_{IL}	Input voltage low	-	-0.4	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:

1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.5.2.3 USB LV Differential DC Characteristics

The DC electrical specifications for the LV Differential signals are described in Table 49-32.

Table 49-32. USB LV Differential DC Characteristics (USBn[1:0], USBp[1:0])

Symbol	Parameter	Min	Max	Unit	I_{OL} / I_{OH}	Notes
V_{DI}	Input Differential Input Sensitivity	0.2		V		1,2, 4
V_{CM}	Input Differential Common Mode Range	0.8	2.5	V		1, 2, 3
V_{SE}	Input Single-Ended Receiver Threshold	0.8	2.0	V		1, 2
V_{CRS}	Output Signal Crossover Voltage	1.3	2.0	V		2
V_{HSSQ}	Input HS Squelch Detection Threshold	100	150	mV		1
V_{HSDSC}	Input HS Disconnect Detection Threshold	525	625	mV		1
V_{HSCM}	Input HS Data Signaling Common Mode Voltage Range	−50	500	mV		1
V_{HSOI}	Output HS Idle Level	−10.0	10.0	mV		1
V_{HSOH}	Output HS Data Signaling High	360	440	mV		1
V_{HSOL}	Output HS Data Signaling Low	−10.0	10.0	mV		1
V_{CHIRPJ}	Output Chirp J Level	700	1100	mV		1
V_{CHIRPK}	Output Chirp K Level	−900	−500	mV		1
V_{OL6}	Output Low Voltage		0.4	V	5 mA	2
V_{OH6}	Output High Voltage	Vcc3_3 − 0.5		V	-2 mA	2
V_{IL4}	Input Low Voltage	−0.5	0.3(Vcc3_3)	V		
V_{IH4}	Input High Voltage	0.5(Vcc3_3)	Vcc3_3 + 0.5	V		

Notes:
1. Applies to High-speed USB 2.0
2. Applies to Low-speed and Full-speed
3. Includes VDI range
4. VDI = USBp[X] - USBn[X]

49.5.5.2.4 USB Clock (CLK48) DC Input Specifications

The DC electrical specifications for the clock input are described in Table 49-33.

Table 49-33. USB Clock (CLK48) DC Input Specifications

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	Vcc3_3 +0.3V	V	-
V_{IL}	Input voltage low	-	-0.4	-	0.8	V	-
I_{leak}	Input Leakage Current	V_{IN}=3.3V	-	-	60	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.5.3 USB AC Characteristics

49.5.5.3.1 USB Transreceiver AC Specifications

The AC electrical specifications of the USB transreceivers in the EP80579 are compliant with the *USB v2.0 specification from April 27, 2000*.

49.5.5.3.2 Transreceiver Timing Waveforms

The timing waveform specifications of the USB Transreceivers in the EP80579 are compliant with the *USB v2.0 specification from April 27, 2000*.

49.5.5.4 USB AC Specifications

Table 49-34. USB Input Clock (CLK48) AC Specifications

Parameters	Min	Typical	Max	Units	Figure
Operating Frequency		48		MHz	-
Frequency Tolerance			100	ppm	-
Duty Cycle	45	-	55	%	-
High Time	7	-	-	ns	49-19
Low Time	7	-	-	ns	49-19
Rise Time	-	-	1.2	ns	49-19
Fall Time	-	-	1.2	ns	49-19

Figure 49-19.Clock Timing

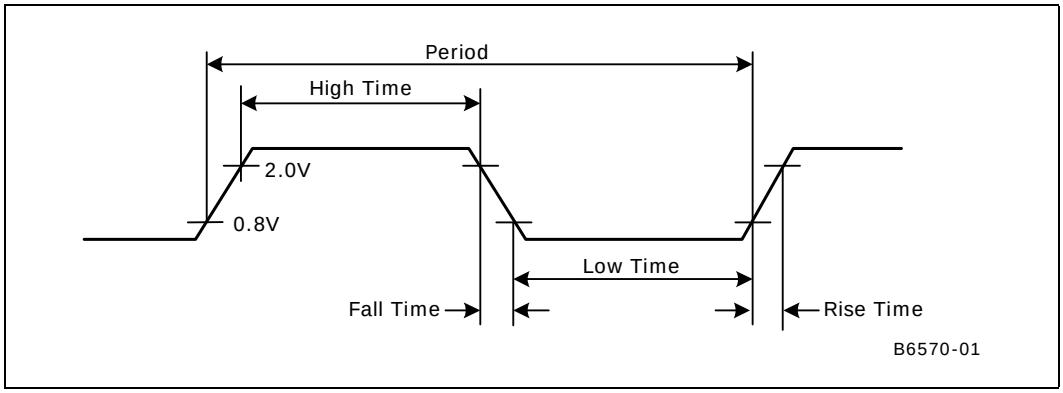


Table 49-35. USB Timing Specifications (Sheet 1 of 2)

Parameter	Min	Max	Units	Figure	Notes
High-speed Source				49-20	7
USBp[x], USBn[x] Driver Rise Time	0.8	1.2	ns	49-20	1, 6 C_L = 10 pF
USBp[x], USBn[x] Driver Fall Time	0.8	1.2	ns		1, 6 C_L = 10 pF
Full-speed Source					8
USBp[x], USBn[x] Driver Rise Time	4	20	ns	49-20	1 C_L = 50 pF
USBp[x], USBn[x] Driver Fall Time	4	20	ns	49-20	1 C_L = 50 pF
Source Differential Driver Jitter: To Next Transition For Paired Transitions	 -3.5 -4	 3.5 4	 ns ns	49-21	2, 3
Source SE0 interval of EOP	160	175	ns		4
Source Jitter for Differential Transition to SE0 Transition	-2	5	ns	49-21	5
Receiver Data Jitter Tolerance: To Next Transition For Paired Transitions	 -18.5 -9	 18.5 9	 ns ns	49-21	3
EOP Width: Must accept as EOP	82		ns	49-22	4
Width of SE0 interval during differential transition		14	ns		
Low-speed Source					9
USBp[x], USBn[x] Driver Rise Time	75	300	ns	49-20	1, 6 C_L = 50 pF C_L = 350 pF
USBp[x], USBn[x] Driver Fall Time	75	300	ns	49-20	1, 6 C_L = 50 pF C_L = 350 pF

Notes:
1. Driver output resistance under steady state drive is specified at 28 ohms at minimum and 43 ohms at maximum.
2. Timing difference between the differential data signals.
3. Measured at crossover point of differential data signals.
4. Measured at 50% swing point of data signals.
5. Measured from last crossover point to 50% swing point of data line at leading edge of EOP.
6. Measured from 10% to 90% of the data signal.
7. High-speed Data Rate has minimum of 479.760 Mb/s and maximum of 480.240 Mb/s
8. Full-speed Data Rate has minimum of 11.97 Mb/s and maximum of 12.03 Mb/s.
9. Low-speed Data Rate has a minimum of 1.48 Mb/s and a maximum of 1.52 Mb/s.

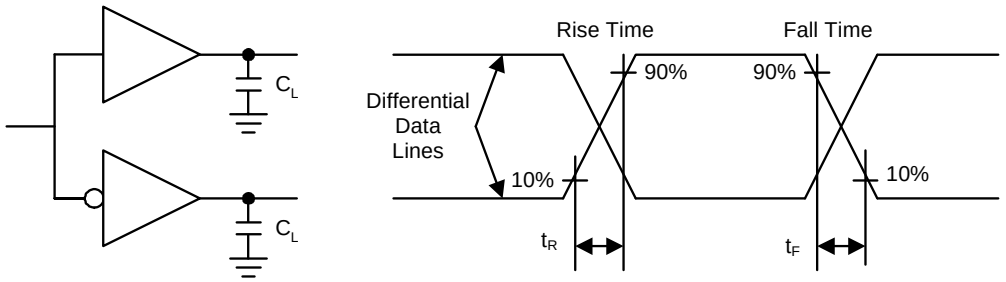
Table 49-35. USB Timing Specifications (Sheet 2 of 2)

Parameter	Min	Max	Units	Figure	Notes
High-speed Source				49-20	7
USBp[x], USBn[x] Driver Rise Time	0.8	1.2	ns	49-20	1, 6 C_L = 10 pF
USBp[x], USBn[x] Driver Fall Time	0.8	1.2	ns		1, 6 C_L = 10 pF
Full-speed Source					8
Source Differential Driver Jitter: To Next Transition For Paired Transitions	 -25 -14	 25 14	 ns ns	49-21	2, 3
Source SE0 interval of EOP	1.25	1.50	µs		4
Source Jitter for Differential Transition to SE0 Transition	-40	100	ns	49-21	5
Receiver Data Jitter Tolerance: To Next Transition For Paired Transitions	 -152 -200	 152 200	 ns ns	49-21	3
EOP Width: Must accept as EOP	670		ns	49-22	4
Width of SE0 interval during differential transition		210	ns		

Notes:

1. Driver output resistance under steady state drive is specified at 28 ohms at minimum and 43 ohms at maximum.
2. Timing difference between the differential data signals.
3. Measured at crossover point of differential data signals.
4. Measured at 50% swing point of data signals.
5. Measured from last crossover point to 50% swing point of data line at leading edge of EOP.
6. Measured from 10% to 90% of the data signal.
7. High-speed Data Rate has minimum of 479.760 Mb/s and maximum of 480.240 Mb/s
8. Full-speed Data Rate has minimum of 11.97 Mb/s and maximum of 12.03 Mb/s.
9. Low-speed Data Rate has a minimum of 1.48 Mb/s and a maximum of 1.52 Mb/s.

Figure 49-20. USB Rise and Fall Times



Low-speed: 75 ns at C_L = 50 pF, 300 ns at C_L = 350 pF
Full-speed: 4 to 20 ns at C_L = 50 pF
High-speed: 0.8 to 1.2 ns at C_L = 10 pF

Figure 49-21.USB Jitter

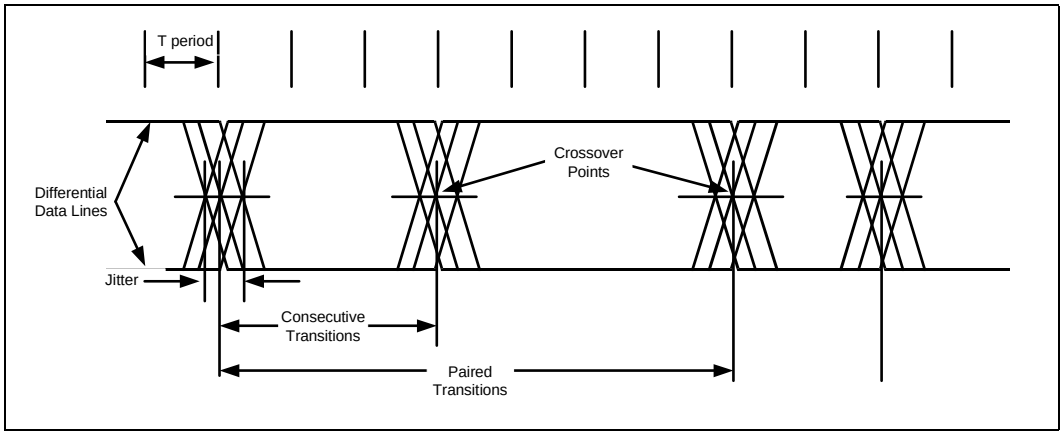
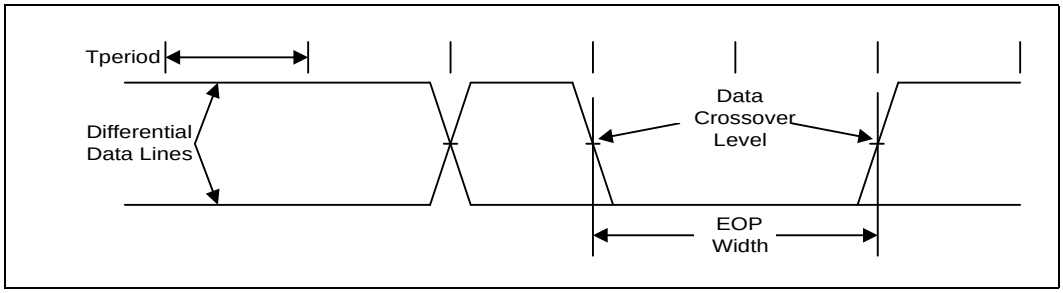


Figure 49-22.USB EOP Width



49.5.6 System Management Bus (SMBus)

The SMBus is SMBus 2.0 compliant and is also compatible with most 2-wire components that are I^2C compatible. The host interface allows the EP80579 to communicate via SMBus, the slave interface allows external microcontrollers to access system resource in the IICH. The SMB does not support access to internal configuration registers. Access to the external DIMMs is through the IICH, via I^2C. This is used to determine the nature of the DIMMs present in order to configure the memory subsystem correctly, including the configuration of the D-unit.

This section describes the electrical characteristics of the System Management Bus (SMBus) interface.

49.5.6.1 SMBus Signal List

For SMBus pin description refer to Table 48-15, “SMBus Interface Signals” on page 1750.

49.5.6.2 SMBus DC Characteristics

Table 49-36. SMBus DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	2
V_{IL}	Input voltage low	-	-	-	0.8	V	2
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	±5	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

1.
2. VIH and VIL for Intruder# is guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-37. SMBus DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OL}	Output voltage low	Iout=4mA			0.4	V	1,2

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.
2. VOL for Intruder# is guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-38. SMBus DC Clock Specification (Sheet 1 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high		2.1	-	3.63	V	-
V_{IL}	Input voltage low			-	0.8	V	-

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-38. SMBus DC Clock Specification (Sheet 2 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
I_{OL}	Output sink current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	±5	µA	-
C_{IN}	-	-		5	-	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.6.3 SMBus AC Characteristics

Table 49-39. SMBus Input AC Characteristics

Sym	Parameter	Min	Typical	Max	Unit	Figure	Notes
f_{SMB}	Bus frequency	10	-	100	kHz		per SMBus specification
T_{su}, t135	SMB_DATA setup time	250	-		ns	49-23	
T_{hd}, t134	SMB_DATA hold time	300	-		ns	49-23	6
T_{DTO}, t136	Device Timeout	25	-	35	ms	49-23	1
t130	Bus Tree Time Between Stop and Start Condition	4.7	-	-	µs	49-23	
t131	Hold Time after (repeated) Start Condition. After this period, the first clock is generated.	4.0	-	-	µs	49-23	
t132	Repeated Start Condition Setup Time	4.7	-	-	µs	49-23	
t133	Stop Condition Setup Time	4.0	-	-	µs	49-23	
t137	Cumulative Clock Low Extend Time (slave device)	-	-	25	ms	49-24	4
t138	Cumulative Clock Low Extend Time (master device)	-	-	10	ms	49-24	5

Notes:
1. A device time outs when any clock low exceeds this parameter.
2. Measured from (V_{IL} -0.15 V) to (V_{IH} +0.15 V).
3. Measured from (V_{IH} +0.15 V) to (V_{IL} -0.15 V), 1 kΩ pull-up, 400 pF load.
4. t137 is the cumulative time a slave device is allowed to extend the clock cycles in one message from the initial start to stop. If a slave device exceeds this time, it is expected to release both its clock and data lines and reset itself.
5. t138 is the cumulative time a master device is allowed to extend its clock cycles within each byte of a message as defined from start-to-ack, ack-to-ack or ack-to-stop.
6. t134 has a minimum timing for I^2C of 0 ns, while the minimum timing for SMBus is 300 ns.

Table 49-40. SMBus Clock Timings (SMBCLK)

Parameter	Conditions	Min.	Typical	Max.	Units	Notes
Operating Frequency		10		100	KHz	
SMB_CLK high time		4.0		50	us	1
SMB_CLK low time		4.7			us	
SMB_CLK Rise time				1000	ns	
SMB_CLK Fall time				300	ns	

Note:
1. The max high time provides a simple guaranteed method for devices to detect bus idle conditions

Table 49-41. SMBus Output AC Characteristics

Sym	Parameter	Min	Typical	Max	Unit	Fig	Notes
t19	Bus frequency	10		100	KHz	49-23	per SMBus specification
t20	Signal rise time			1000	ns	49-23	2
t21	Signal fall time			300	ns	49-23	3

Notes:
1. A device times out when any clock low exceeds this parameter.
2. Measured from (V_{IL} -0.15 V) to (V_{IH} +0.15 V).
3. Measured from (V_{IH} +0.15 V) to (V_{IL} -0.15 V), 1 kΩ pull-up, 400 pF load.

Figure 49-23. SMBus Transaction

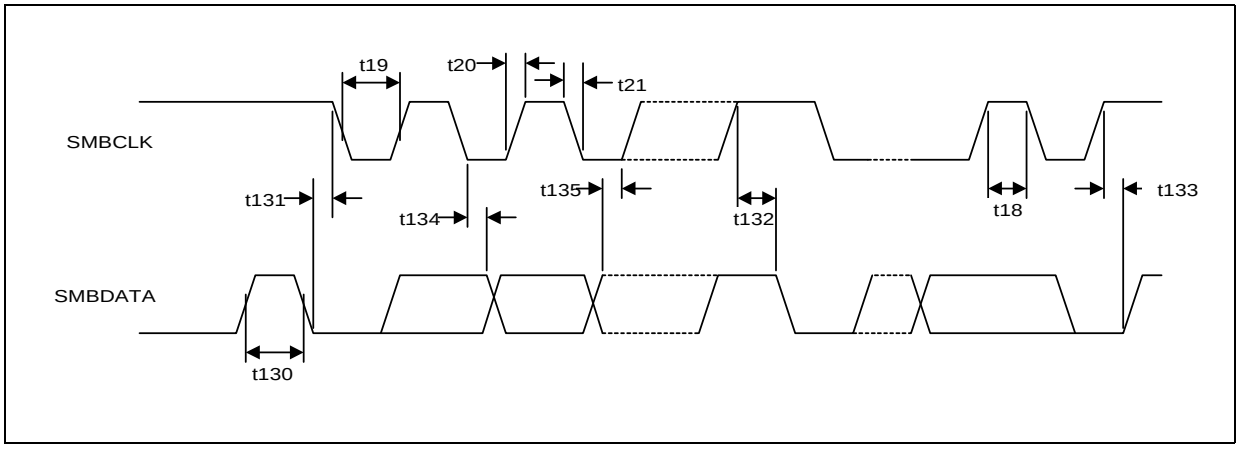
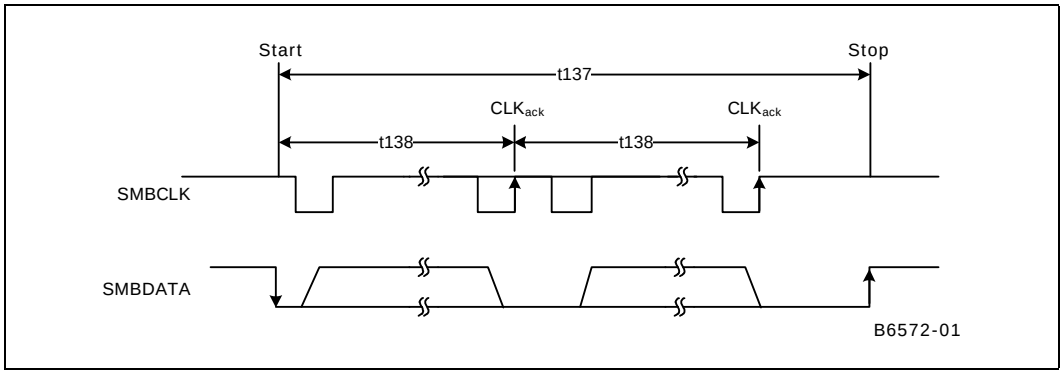


Figure 49-24.SMBus Timeout



49.5.7 UART

This section describes the Universal Asynchronous Receiver/Transmitter (UART) serial port used for the two UARTs that are integrated into the Serial I/O unit and Watchdog Timer (SIW). The UART can be controlled via programmed I/O. The basic programming model is the same for both UARTs, with the only difference being the Logical Device Number assigned to each. The serial port consists of a UART that supports all the functions of a standard 16550 UART, including hardware flow control interface.

This section describes the electrical characteristics of the UART interface.

49.5.7.1 UART Signal List

For UART pin description refer to Table 48-16, “UART Signals” on page 1751.

49.5.7.2 UART DC Characteristics

Table 49-42. UART DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	VCC33 +0.5	V	-
V_{IL}	Input voltage low	-	-0.5	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-43. UART DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-0.5mA	0.9VCC33	-	-	V	1
V_{OL}	Output voltage low	I_{out}=1.5mA	-	-	0.1VCC33	V	
I_{OH}	Output current at high voltage	$V_{OH}>0.9VCC33$	-0.5	-	-	mA	
I_{OL}	Output current at low voltage	$V_{OL}<0.1VCC33$	-	-	1.5	mA	

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

Table 49-44. UART DC Clock Specification

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	0.5VCC33	-	VCC33 + 0.5	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<VCC33$	-10	-	10	µA	-
C_{IN}	Input-pin capacitance	-	5	-	12	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.7.3 UART AC Characteristics

The AC electrical specifications of the UART interface are compliant with the *PCI Local Bus Specification*, Rev 3.0 (refer to AC specifications for 3.3V signaling for more information).

49.5.7.4 UART Receiver AC Specifications

Table 49-45. UART Timing

Parameter	Min.	Typical	Max.	Units	Notes
SIU_RXD[2:1], SIU_CTS[2:1]#, SIU_DSR[2:1]#, SIU_DCD[2:1]#, and SIU_RI[2:1]# Set up time to UART_CLK rising	7			ns	
SIU_RXD[2:1], SIU_CTS[2:1]#, SIU_DSR[2:1]#, SIU_DCD[2:1]#, and SIU_RI[2:1]# hold time from UART_CLK rising	0			ns	

49.5.7.4.1 UART Transmitter AC Specifications

Table 49-46. UART Timing

Parameter	Min.	Typical	Max.	Units	Notes
SIU_TXD[2:1] Valid Delay from UART_CLK rising	2		11	ns	
SIU_TXD[2:1],SIU_DTR[2:1]# and SIU_RTS[2:1]#, Valid Delay from UART_CLK rising	2			ns	

49.5.7.4.2 UART Clock Timings

14.7456 MHz, and 48 MHz are supported for UART baud clock input.

49.5.8 Serial Peripheral Interface (SPI)[1]

This section describes the electrical characteristics of the Serial Peripheral Interface.

49.5.8.1 SPI Signal List

For SPI pin description refer to Table 48-13, “SPI Interface Signals” on page 1749.

49.5.8.2 SPI DC Characteristics

Table 49-47. SPI DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-48. SPI DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	1
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.8.3 SPI AC Characteristics

Table 49-49. SPI Timing Specifications (Sheet 1 of 2)

Symbol	Parameter	Min	Max	Units	Figures	Notes
	Serial Clock Period - 20 MHz Operation	17.2	18.4	MHz	-	1
t182	SPI Clock Duty cycle at the host	-	50	%	49-25	-
t183	Tco of SPI_MOSI with respect to serial clock falling edge at the host	-5	13	ns	49-25	-

Notes:
1. The typical clock frequency driven by the EP80579 is 17.86 MHz.

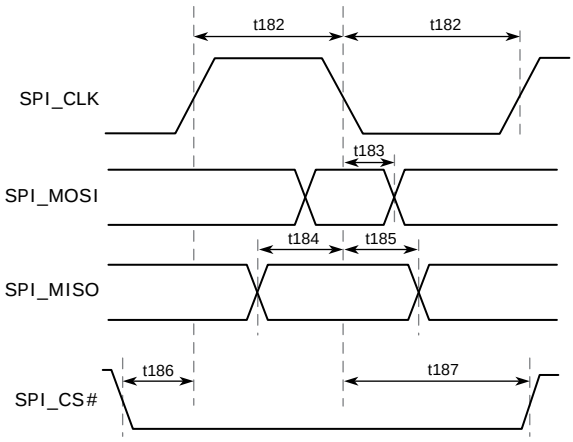
1. Intel recommends using the SPI for Pre-boot firmware due to the reduced availability of LPC FWH.

Table 49-49. SPI Timing Specifications (Sheet 2 of 2)

Symbol	Parameter	Min	Max	Units	Figures	Notes
t184	Setup of SPI_MISO with respect to serial clock falling edge at the host	16	-	ns	49-25	-
t185	Hold of SPI_MISO with respect to serial clock falling edge at the host	0	-	ns	49-25	-
t186	Setup of SPI_CS[1:0]# assertion with respect to serial clock rising at the host	30	-	ns	49-25	-
t187	Hold of SPI_CS[1:0]# deassertion with respect to serial clock falling at the host	30	-	ns	49-25	-

Notes:
1. The typical clock frequency driven by the EP80579 is 17.86 MHz.

Figure 49-25. SPI Timing Diagram



49.5.9 Low Pin Count (LPC)

The LPC interface is used to control all the logical blocks on the Serial I/O unit and Watchdog Timer (SIW).LPC bus signals use PCI 33 MHz electrical signal characteristics. Refer to the *Low Pin Count (LPC) Interface Specification, Rev 1.1* for more information.

This section describes the electrical characteristics of the LPC interface.

49.5.9.1 LPC Signal List

For LPC pin description refer to Table 48-14, “LPC and FWH Interface Signals” on page 1749.

49.5.9.2 LPC DC Characteristics

Table 49-50. LPC DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	VCC33 +0.5	V	-
V_{IL}	Input voltage low	-	-0.5	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-51. LPC DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-0.5mA	0.9VCC33	-	-	V	1
V_{OL}	Output voltage low	I_{out}=1.5mA	-	-	0.1VCC33	V	
I_{OH}	Output current at high voltage	$V_{OH}>0.9VCC33$	-0.5	-	-	mA	
I_{OL}	Output current at low voltage	$V_{OL}<0.1VCC33$	-	-	1.5	mA	

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

Table 49-52. LPC DC Clock Specifications

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	0.5VCC33	-	VCC33 + 0.5	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<VCC33$	-10	-	10	µA	-
C_{IN}	Input-pin capacitance	-	5	-	12	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.9.3 LPC AC Characteristics

Table 49-53. LPC Clock AC Characteristics

Parameter	Min	Max	Units	Figure
Period	30	33.3	ns	49-26
High Time	11	-	ns	49-26
Low Time	11	-	ns	49-26
Rise Time	-	3	ns	49-26
Fall Time	-	3	ns	49-26

49.5.9.3.1 LPC Timing Specification

Table 49-54. LPC Input Timing Specification

Parameter	Min	Typical	Max	Units	Figure
LAD[3:0]/FWH[3:0] Setup Time to PCICLK Rising	7			ns	49-27
LAD[3:0/FWH[3:0]] Hold Time from PCICLK Rising	0			ns	49-27
LDRQ[1:0]# Setup Time to PCICLK Rising	12			ns	49-27
LDRQ[1:0]# Hold Time from PCICLK Rising	0			ns	49-27

Table 49-55. LPC Output Timing Specification

Parameter	Min	Typical	Max	Units	Figure
LAD[3:0]/FWH[3:0] Valid Delay from PCICLK Rising	2		11	ns	49-28
LAD[3:0]/FWH[3:0] Output Enable Delay from PCICLK Rising	2			ns	49-29
LAD[3:0]/FWH[3:0] Float Delay from PCICLK Rising			28	ns	49-30
LFRAME#/FWH[4] Valid Delay from PCICLK Rising	2		11	ns	49-28

49.5.9.3.2 LPC Timing Diagrams

Figure 49-26. LPC Clock (PCICLK) Timing Diagram

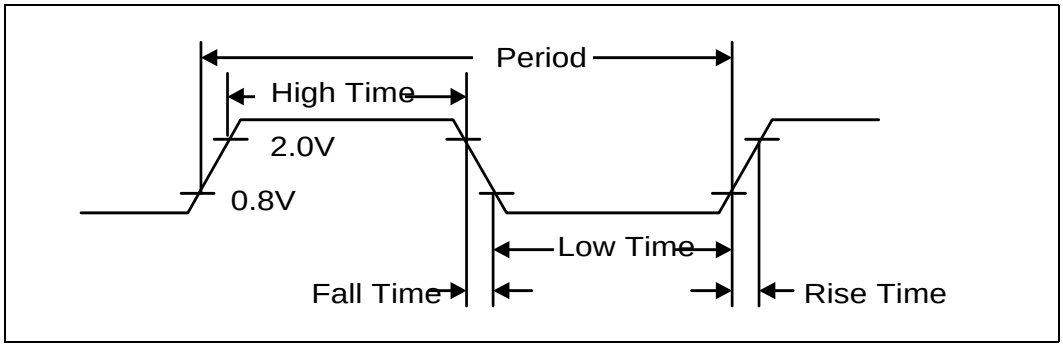


Figure 49-27.LPC Input Setup and Hold Timing Diagram

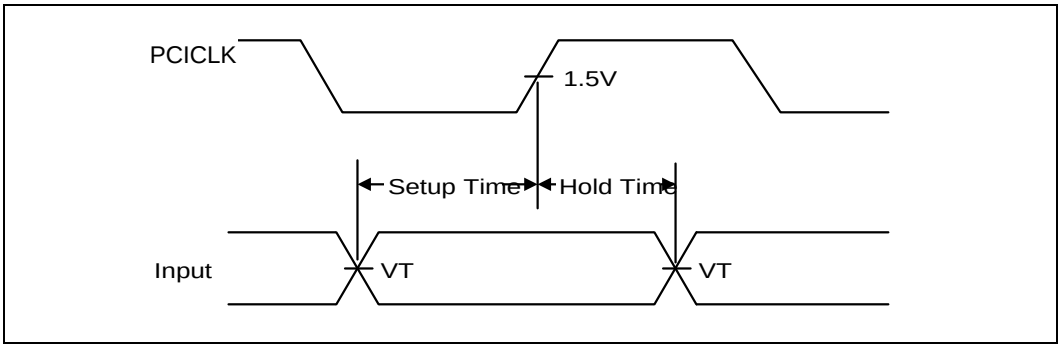


Figure 49-28.LPC Valid Delay from Rising Clock Edge Diagram

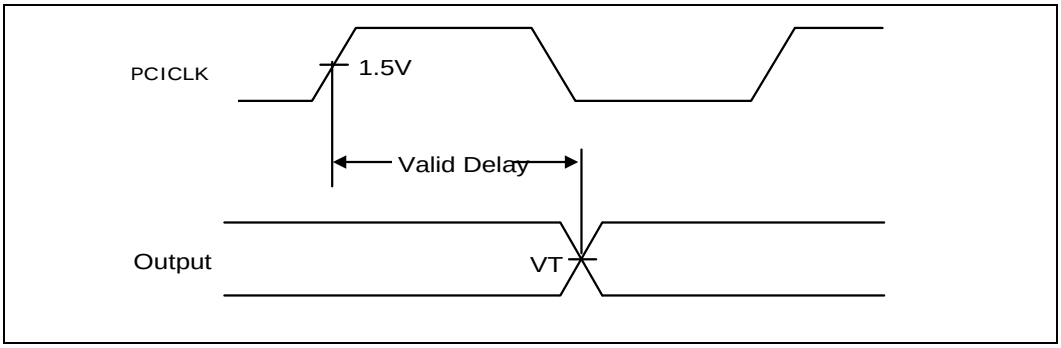


Figure 49-29.LPC Output Enable Delay Diagram

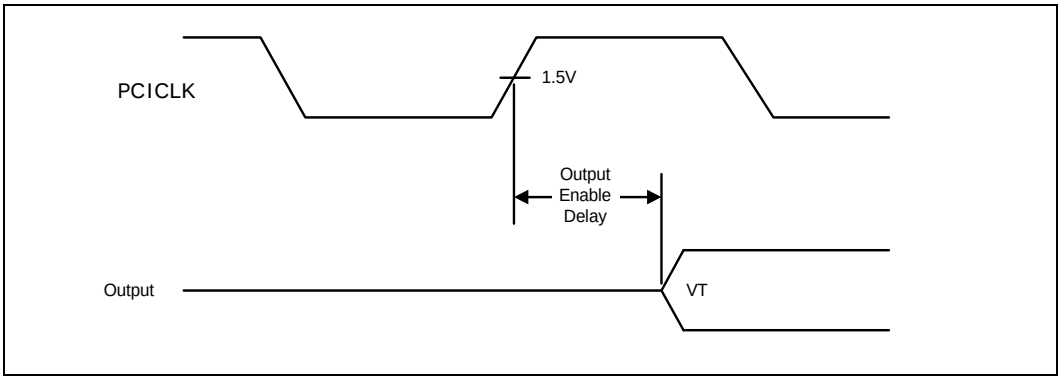
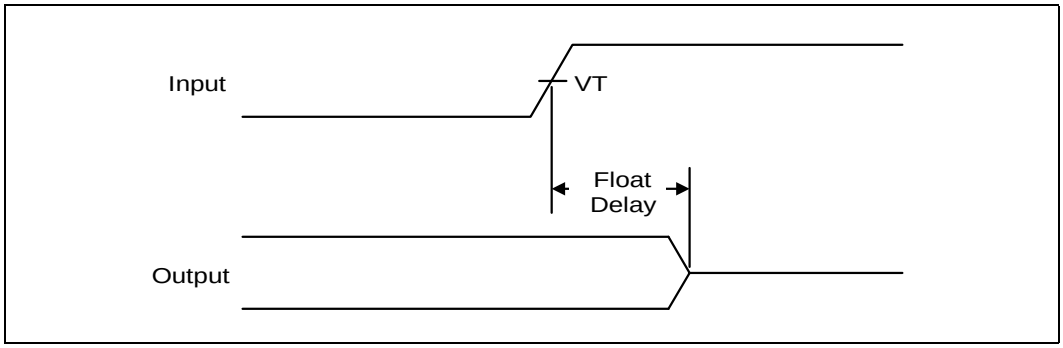


Figure 49-30.LPC Float Delay Diagram



49.5.9.3.3 LPC Reset

The LPC controller does not adhere to the section of the LPC specification that says “LRESET# is always asserted after LPCPD#”. LRESET # is not always asserted after LPCPD#.

49.5.10 General Purpose I/O (GPIO)

GPIO are general-purpose I/O signals. These signals can be used for slow-speed, software-controlled I/O.

This section describes the electrical characteristics of the GPIO interface.

49.5.10.1 GPIO Signal List

For GPIO pin description refer to Table 48-11, “General-Purpose IO Signals” on page 1744.

49.5.10.2 GPIO DC Characteristics

Table 49-56.GPIO DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	VCC33 +0.5	V	-
V_{IL}	Input voltage low	-	-0.5	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing

Table 49-57. GPIO DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-0.5mA	0.9VCC 33	-	-	V	1
V_{OL}	Output voltage low	I_{out}=1.5mA	-	-	0.1VCC 33	V	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.10.3 GPIO AC Specifications

The AC electrical specifications of the GPIO interface are compliant with the *PCI Local Bus Specification, Rev. 3.0* (refer to the AC Specifications for 3.3V Signaling for more information).

49.5.11 IICH Interrupt Signal

This section describes the electrical characteristics of the IICH interrupt signal interface.

49.5.11.1 IICH Interrupt Signal List

For IICH interrupt signal pin descriptions, refer to Table 48-12, “IICH Interrupt Signals” on page 1748.

49.5.11.2 IICH Interrupt Signal DC Characteristics

Table 49-58. IICH Interrupt Signal DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-59. IICH Interrupt Signal DC Output Characteristics (Sheet 1 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.4	-		V	1

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

Table 49-59. IICH Interrupt Signal DC Output Characteristics (Sheet 2 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OL}	Output voltage low	I_{out}=6mA		-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-	-	1.5	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	-	-	-0.5	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.11.3 IICH Interrupt Signal AC Input, Output Characteristics

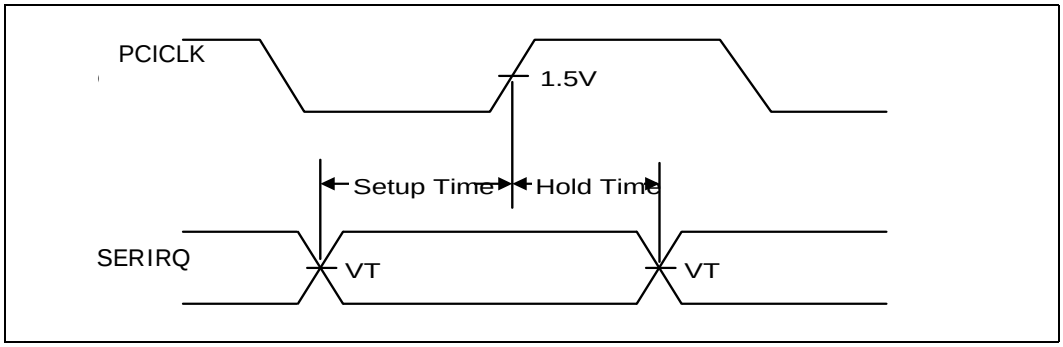
The AC electrical specifications of the IICH Interrupt Signal are compliant with the *PCI Local Bus Specification, rev.* 3.0 (refer to AC Specifications for 3.3V Signaling for more information).

49.5.11.4 IICH Interrupt Signal Timing Specification

Table 49-60. IICH Interrupt Signal Timing Specification

Parameter	Min	Typical	Max	Units	Note
SERIRQ Setup Time to PCICLK Rising	7			ns	49-31
SERIRQ Hold Time from PCICLK Rising	0			ns	49-31

Figure 49-31. IICH Interrupt Signal Timing Diagram



49.5.11.5 IICH Clock AC Specifications

Table 49-61. IICH Clock (CLK14) AC Specifications (Sheet 1 of 2)

Parameters	Min	Typical	Max	Units	Figure	Note
Operating Frequency		14.31818		MHz	-	
Period	67	-	70	ns		
High Time	20	-	-	ns	49-32	

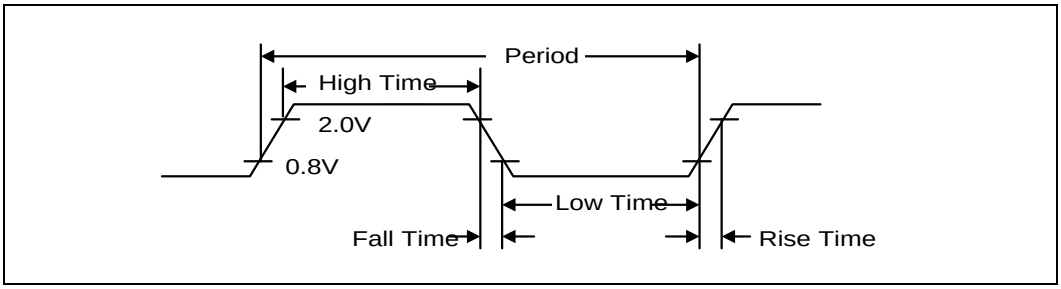
Notes:
1. CLK14 edge rates in a system as measured from 0.8 V to 2.0 V.

Table 49-61. IICH Clock (CLK14) AC Specifications (Sheet 2 of 2)

Parameters	Min	Typical	Max	Units	Figure	Note
Low Time	20	-	-	ns	49-32	
Rising Edge Rate	1.0	-	4.0	V/ns	49-32	1
Falling Edge Rate	1.0	-	4.0	V/ns	49-32	1

Notes:
1. CLK14 edge rates in a system as measured from 0.8 V to 2.0 V.

Figure 49-32. IICH Clock (CLK14) Timing Diagram



49.5.12 Real Time Clock (RTC)

This section describes the electrical characteristics of the Real Time Clock interface.

49.5.12.1 RTC Signal List

For Real Time Clock pin description refer to Table 48-10, “Real Time Clock Interface Signals” on page 1744.

49.5.12.2 RTC DC Characteristics

Table 49-62. RTC DC Input Characteristics (RTEST#)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	6	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-63. RTC DC Clock Input Characteristics (RTCX[2:1])

Symbol	Parameter	Min	Typical	Max	Units
V_{IL}	Input voltage low	-	-	0.10	V
V_{IH}	Input voltage high	0.40	-	1.2	V
C_L	RTCX1 typical value	6			pf
C_L	RTCX2 typical value	6			pf

Notes:
1. 3.3 V Clock Input
2. Guaranteed by design.

49.5.12.3 RTC AC Characteristics

Table 49-64. RTC Clock Input (RTCX[2:1])Timing Values

Symbol	Parameter	Min	Nominal	Max	Units	Notes
RTCX[2:1]	RTC clock frequency		32.768		KHz	1, 2
Tolerance	This parameter specifies the crystal tolerance for RTCX[2:1]	-20	-	+20	ppm	

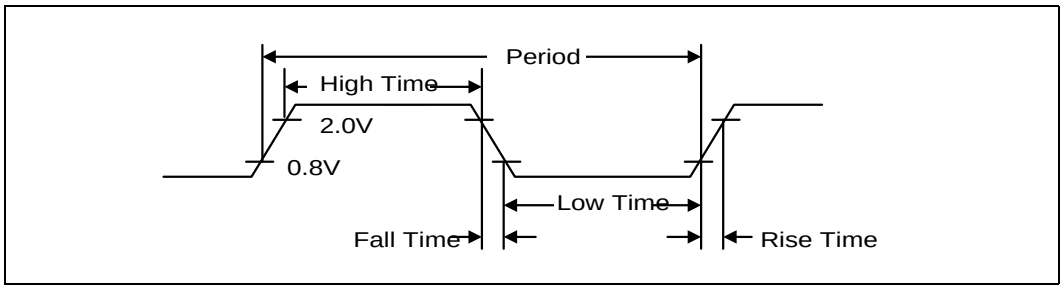
Notes:
1. RTCX[1] can be connected to a crystal along with RTCX[2] or to an oscillator
2. When using an oscillator only RTCX[1] is connected and RTCX[2] can remain unconnected.

Table 49-65. RTC Clock Output (SUSCLK) Timings

Parameters	Min	Typical	Max	Units	Figure	Note
Operating Frequency	-	32.768	-	kHz	-	1
Duty Cycle	30	-	70	%		
High Time	10	-	-	us	49-33	
Low Time	10	-	-	us	49-33	

Notes:
1. SUSCLK is the RTC generator output

Figure 49-33. RTC Clock Output (SUSCLK) Timing Diagram



49.5.13 Gigabit Ethernet (GbE: RMII, RGMII, MDIO, EEPROM)

The EP80579 contains three Gigabit Ethernet MACs. The GbE Ethernet MACs contained in the EP80579 resemble the Intel® 82545 Gigabit Ethernet Controller.

The EP80579 Gigabit Ethernet (GbE) MAC is based on a fourth generation Gigabit MAC to provide a standard IEEE 802.3 Ethernet interface for 1000BASE-T, 100BASE-Tx, and 10BASE-T applications. The MAC is capable of transmitting and receiving data rates of 10/100/1000 Mbps. Through the GbE interfaces, the EP80579 can be connected to Reduced Ethernet (RMII), or Reduced GMII (RGMII) external PHY devices.

The EP80579 can support both 3.3V and 2.5V PHY for the GbE interface. The media outputs use 2.5V drivers that are 3.3V tolerant. This is a violation of the RMII specification, which calls for TTL level outputs (not LVTTL) and 5V input level tolerance. All major PHY devices that may be used in an EP80579 system are expected to drive a maximum voltage of 3.3V.

RGMII version 1.3 is specified as a 2.5V CMOS level I/O interface and as such, the EP80579 is fully compliant. RGMII version 2.0 is specified as a 1.5 V HSTL I/O interface and the EP80579 is NOT compliant with this standard. All PHY devices that are used with the EP80579 must support version 1.3. The MDIO, MDC, and serial EEPROM bus are implemented with 2.5V I/O drivers.

Management Data Input/Output (MDIO) interface provides a path to transfer control information and status between the PHY and the EP80579. The MDIO interface allows the software to continuously poll the PHY's configuration registers through and reprogram the GbE configuration. Management Data Clock (MDC) is sourced by the EP80579 to the PHY devices as a timing reference for the MDIO interface.

A single four-wire Microwire* interface is provided for connection of an optional, externally connected serial EEPROM. The serial EEPROM may be used to provide configuration information to the GbE MACs upon power-up or reset. All three MACs share the same EEPROM.

This section describes the electrical characteristics of the GbE interface.

49.5.13.1 Gigabit Ethernet Signal List

For Gigabit Ethernet MAC pin description refer to Table 48-22, "Gigabit Ethernet Interface Signals" on page 1760.

49.5.13.2 Gigabit Ethernet DC Characteristics

49.5.13.2.1 DC Characteristics: RMII and RGMII Mode of Operation

Table 49-66. DC Input Characteristics: RMII Mode of Operation (Sheet 1 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input high voltage	$V_{IH} > V_{IH\_Min}$ V_{CC} = Min	2.0	-	3.6	V	-
V_{IL}	Input low voltage	$V_{IH} > V_{IL\_Max}$ V_{CC} = Min	-	-	0.8	V	-
I_{IH}	Input high current	V_{CC} = Max V_{IN} = 2.5 V	-	-	15	uA	-

Table 49-66. DC Input Characteristics: RMII Mode of Operation (Sheet 2 of 2)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
I_{IL}	Input low current	V_{CC} = Max V_{IN} = 0.4 V	-15	-	-	uA	-
C_{IN}	Input pin cap		-	-	8	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-67. DC Output Characteristics: RMII Mode of Operation

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output high voltage	V_{CC} = 2.5V I_{OH} = -1mA	2.4	-	See Notes	V	1,Note 2
V_{OH}	Output high voltage	V_{CC} = 2.375V I_{OH} = -1mA	2.2	-	See Notes	V	1,Note 2
V_{OL}	Output low voltage	V_{CC} = Min I_{OL} = 1mA	-	-	0.4	V	

Notes:
1. The RMII buffer is powered from the 2.5v power rail - the output signal level is 100mV below the rail for RMII. In the ideal case, the Voh equals the 2.4v in accordance to the RMII specification. If the 2.5v power rail drops below the ideal value the buffer will violate the RMII specification and the 0.4v margin between Voh/Vih will be reduced under this condition.
2. V_{OH} Max = VCC25 for GbE Ports 1 and 2 or VCCSUS25 for GbE Port 0

Table 49-68. DC Input Characteristics: RGMII Mode of Operation

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input high voltage	$V_{IH} > V_{IH\_Min}$ V_{CC} = Min	1.7	-	3.6	V	-
V_{IL}	Input low voltage	$V_{IH} > V_{IL\_Max}$ V_{CC} = Min	-	-	0.7	V	-
I_{IH}	Input high current	V_{CC} = Max V_{IN} = 2.5 V	-	-	15	uA	-
I_{IL}	Input low current	V_{CC} = Max V_{IN} = 0.4 V	-15	-	-	uA	-
C_{IN}	Input pin cap		-	-	8	pF	Note 1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-69. DC Output Characteristics: RGMII Mode of Operation

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output high voltage	V_{CC} = 2.5V I_{OH} = -1mA	2.4	-	See Notes	V	1
V_{OL}	Output low voltage	V_{CC} = Min I_{OL} = 1mA	-	-	0.4	V	

Notes:
1. V_{OH} Max = VCC25 for GbE Ports 1 and 2 or VCCSUS25 for GbE Port 0

Table 49-70. DC Input Characteristics: MDIO Mode of Operation

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input high voltage	$V_{IH} > V_{IH\_Min}$ V_{CC} = Min	2.0	-	3.6	V	-
V_{IL}	Input low voltage	$V_{IH} > V_{IL\_Max}$ V_{CC} = Min	-	-	0.8	V	-
I_{IH}	Input high current	V_{CC} = Max V_{IN} = 2.5 V	-	-	15	uA	-
I_{IL}	Input low current	V_{CC} = Max V_{IN} = 0.4 V	-15	-	-	uA	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc2\_5$	-10	-	10	µA	-
C_{IN}	Input pin cap		-	-	8	pF	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-71. DC Output Characteristics: MDIO Mode of Operation

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output high voltage	V_{CC} = 2.5V I_{OH} = -1mA	2.4	-	=VCCSUS25	V	1
V_{OH}	Output high voltage	V_{CC} = 2.375V I_{OH} = -4mA	2.15	-	=VCCSUS25	V	1
V_{OL}	Output low voltage	V_{CC} = Min I_{OL} = 1mA	-	-	0.4	V	

Notes:
1. The MDIO buffer is powered from the 2.5v power rail - the output signal level is 200mV below the rail for MDIO. In the ideal case, the Voh equals the 2.4v in accordance to the RMII specification. If the 2.5v power rail drops below the ideal value the buffer will violate the RMII specification and the 0.4v margin between Voh/Vih will be reduced under this condition.

49.5.13.2.2 DC Characteristics: EEPROM Interface

Table 49-72. DC Input Characteristics: EEPROM Interface

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input high voltage	$V_{IH} > V_{IH\_Min}$ V_{CC} = Min	1.7	-	3.6	V	-
V_{IL}	Input low voltage	$V_{IH} > V_{IL\_Max}$ V_{CC} = Min	-	-	0.7	V	-
C_{IN}	Input pin cap		-	-	8	pF	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-73. DC Output Characteristics: EEPROM Interface

Symbol	Parameter	Conditions	Min	Typical	Max	Units
V_{OH}	Output high voltage	V_{CC} = 2.5V I_{OH} = -1mA	2.4	-	=VCCSUS25	V
V_{OL}	Output low voltage	V_{CC} = Min I_{OL} = 1mA	-	-	0.4	V

49.5.13.2.3 GbE DC Clock Specifications

Table 49-74. Reference Clock DC Input Specification: (GBE_REFCLK_RMII, GBE_REFCLK)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input high voltage	$V_{IH} > V_{IH\_Min}$ V_{CC} = Min	1.7	-	3.6	V	-
V_{IL}	Input low voltage	$V_{IH} > V_{IL\_Max}$ V_{CC} = Min	-	-	0.7	V	-
I_{IH}	Input high current	V_{CC} = Max V_{IN} = 2.5 V	-	-	15	uA	-
I_{IL}	Input low current	V_{CC} = Max V_{IN} = 0.4 V	-15	-	-	uA	-
C_{IN}	Input pin cap		-	-	8	pF	Note 1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.13.3 Gigabit Ethernet AC Characteristics

49.5.13.3.1 Frequency Requirements

Table 49-75 provides the frequency of all input clocks in a different mode of operation.

Table 49-75. Frequencies of All Input Clocks

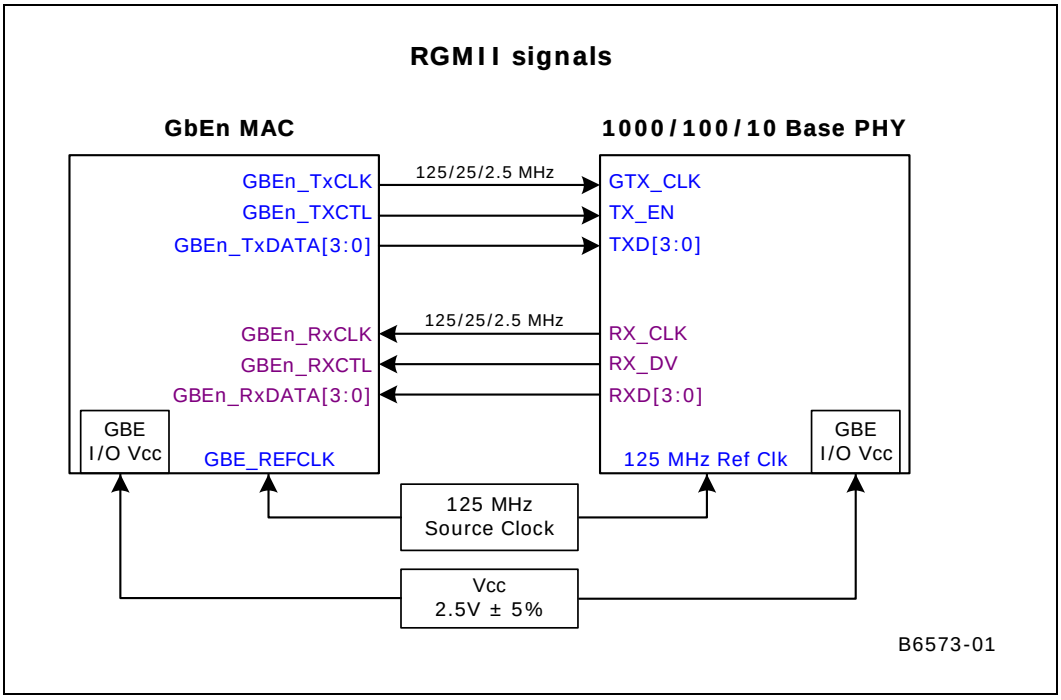
Model	Clock Name	Frequency
RMII	GBE_REFCLK_RMII	50 MHz
RGMII	GBEn_RxCLK	2.5/25/125 MHz
	GBEn_TxCLK	2.5/25/125 MHz
	GBE_REFCLK	125 MHz

49.5.13.3.2 GbE Ethernet Interface — RGMII Mode

This section describes the GbE Ethernet reference clock transmit and receive timings when operating in RGMII mode. RGMII works in source synchronous mode, where transmission and receive are done with respect to TXCLK and RXCLK. In RGMII mode, TXCLK and RXCLK run at 2.5/25/125MHz. In 2.5/25MHz mode, both transmit and receive occurs on rising edges of the clock. At 125MHz, data transfer (both Tx and Rx) occurs on both edges of the clock (same as DDR2).

The RGMII interface gets Rx clock from the PHY and Tx clock from the MAC. Both MAC and PHY receive a 125MHz reference clock from the board. The RGMII outputs uses 2.5V drivers that are 3.3V tolerant. Figure 49-34 shows the GbE block diagram with the signal connections in RGMII mode.

Figure 49-34.GbE RGMII Mode Signal Connection Block Diagram



49.5.13.3.3 GbE Ethernet Interface — RGMII Mode Reference Clock

Table 49-76 shows the RGMII 125 MHz reference clock AC specifications.

Table 49-76. GbE RGMII Reference Clock Timing Values

Symbol	Parameter	Min	Nominal	Max	Units	Notes
T_1	RGMII reference clock period	-	8.0	-	ns	1
	Frequency accuracy	-50	-	+50	ppm	1
T_2	Duty cycle with respect to T_1 (Nom.)	45	50	55	%	1, 5
T_3	Low to high rise time (20% to 80%)	-	-	0.75	ns	2, 5
T_4	High to low fall time (80% to 20%)	-	-	0.75	ns	3, 5
T_{jitter}	Cycle to Cycle Jitter	-	-	80	ps	4, 5
	Peak to peak Jitter (aggregate)	-	-	150	ps	4, 5

Notes:
1. Applies to GBEn_REFCLK signal operating in RGMII mode.
2. Measurement points for Rise time are 20% GBE V_{CC} to 80% GBE V_{CC}.
3. Measurement points for Fall time are 80% GBE V_{CC} to 20% GBE V_{CC}.
4. The maximum allowable jitter is 80 ps. This jitter can be less but never greater than 80 ps. This reference clock jitter directly translates into the jitter produced on the GBEn_TxCLK signal. For example, a 40 ps jitter on the reference clock input would generate the same 40 ps jitter on the GBEn_TxCLK signal.
5. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Figure 49-35 shows the RGMII 125 MHz reference clock timing diagram.

49.5.13.3.4 GbE Ethernet Interface — RGMII Mode Transmit Timings

Figure 49-35. RGMII 125 MHz Reference Clock Timing Diagram

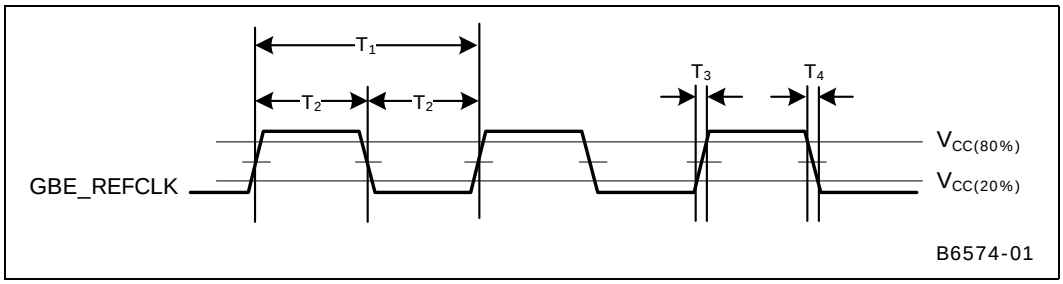


Table 49-76 shows the AC specifications.

Table 49-77. GbE Transmit Timing Values — RGMII Mode

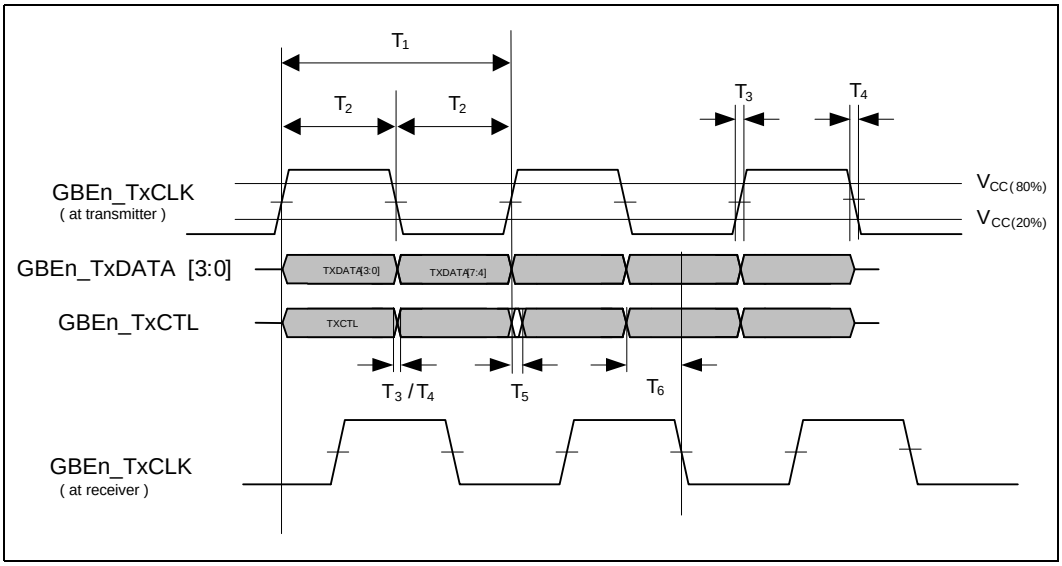
Symbol	Parameter	Min	Nominal	Max	Units	Notes
T_1	RGMII output clock period (1000 Base)	7.2	8.0	8.8	ns	1, 7
T_1	RGMII output clock period (100 Base)	36	40	44	ns	1, 7
T_1	RGMII output clock period (10 Base)	360	400	440	ns	1, 7
T_2	1000 base duty cycle with respect to T_1 (Nom.)	45	50	55	%	1, 3, 7
T_2	100/10 base duty cycle with respect to T_1 (Nom.)	40	50	60	%	1, 3, 7
T_3	Low to high rise time (20% to 80%)	-	-	0.75	ns	1, 4, 7
T_4	High to low fall time (80% to 20%)	-	-	0.75	ns	1, 5, 7
T_5	Data to clock output SKEW (at transmitter)	-500	0	500	ps	2, 7
T_6	Data to clock output SKEW (at receiver)	1	-	2.6	ns	6, 7

Notes:

1. This applies to the GBEn_TxCLK signal operating in RGMII mode.
2. This applies to the GBEn_TXDATA[3:0] signals for RGMII mode.
3. Duty cycle may be stretched/shrunk during speed changes or while transitioning to a received packet's clock domain as long as minimum duty cycle is not violated and stretching occurs for no more than three Tcyc of the lowest speed transitioned between.
4. Measurement points for Rise time are 20% GBE V_{CC} to 80% GBE V_{CC}.
5. Measurement points for Fall time are 80% GBE V_{CC} to 20% GBE V_{CC}.
6. The RGMII specification v1.3 requires the board design to induce trace delay such that an additional trace delay of greater than 1.5 ns and less than 2.0 ns is added to the GBEn_TxCLK signal between the MAC output and the PHY input. This method would require "x-inches" of trace to routed for the GBEn_TxCLK signal. Some PHYs solve this by supporting an internal delay within the PHY. Refer to your chosen PHY data sheet for additional information on these clock delay modes.
7. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Figure 49-36 shows the RGMII transmit timing diagram.

Figure 49-36.GbE Transmit Waveform — RGMII Mode



49.5.13.3.5 GbE Ethernet Interface — RGMII Mode Receive Timings

Table 49-78 shows the AC specification.

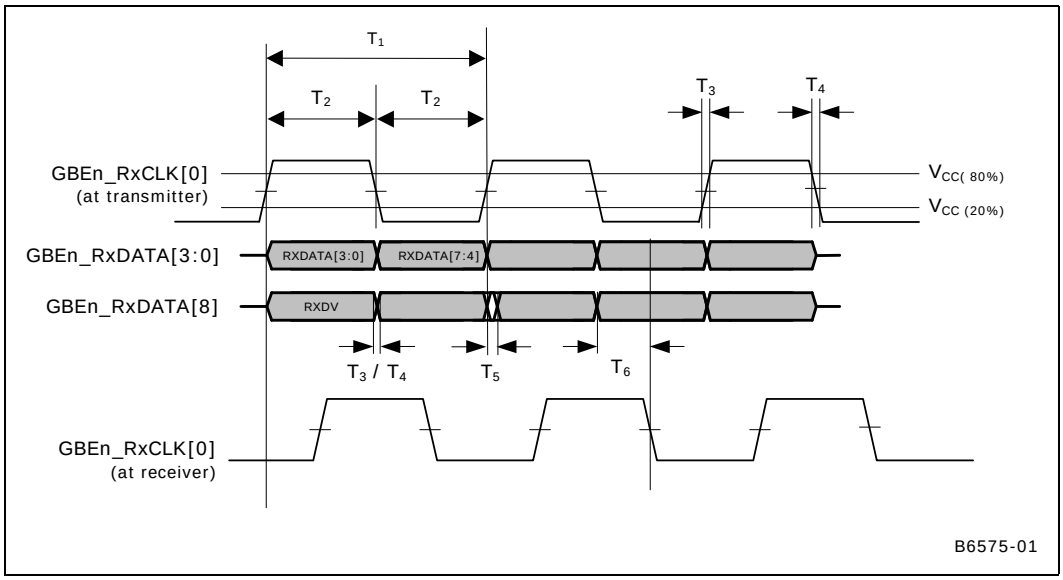
Table 49-78. GbE Receive Timing Values — RGMII Mode

Symbol	Parameter	Min	Nominal	Max	Units	Notes
T_1	RGMII input clock period (1000 Base)	7.2	8.0	8.8	ns	1, 6
T_1	RGMII input clock period (100 Base)	36	40	44	ns	1, 6
T_1	RGMII input clock period (10 Base)	360	400	440	ns	1, 6
T_2	1000 base duty cycle with respect to T_1 (Nom.)	45	50	55	%	1, 3, 6
T_2	100/10 base duty cycle with respect to T_1 (Nom.)	40	50	60	%	1, 3, 4, 6
T_3	Low to high rise time (20% to 80%)	-	-	1.0	ns	1, 5, 6
T_4	High to low fall time (80% to 20%)	-	-	1.0	ns	1, 5, 6
T_5	Data to clock output SKEW (at transmitter)	-500	0	500	ps	1, 3, 6
T_6	Data to clock output SKEW (at receiver)	1	-	2.6	ns	6

Notes:
1. This applies to the GBEn_RxCLK signal operating in RGMII mode.
2. This applies to the GBEn_RXDATA[3:0] signals for RGMII mode.
3. Duty cycle may be stretched/shrunk during speed changes or while transitioning to a received packet's clock domain as long as minimum duty cycle is not violated and stretching occurs for no more than three Tcyc of the lowest speed transitioned between.
4. Measurement points for Rise time are 20% GBE V_{CC} to 80% GBE V_{CC}.
5. Measurement points for Fall time are 80% GBE V_{CC} to 20% GBE V_{CC}.
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Figure 49-37 shows the RGMII receive timing diagram.

Figure 49-37. GbE Receive Waveform — RGMII Mode



49.5.13.3.6 Design to Test Loads

The I/Os are implemented to support a 5pF load to ground for AC timing.

49.5.13.3.7 GbE Ethernet Interface — RMII Mode

This section describes the AC timings for GbE interface when the interconnect is designed for a RMII interface.

In RMII mode of operation, both transmit and receive (TX_DATA and RX_DATA) are with respect to external clock, which in 10/100 Base T is 50 MHz.

Figure 49-38 shows the GbE signal connections used in RMII mode for 10/100 Base connections. An external clock is sourcing the reference clock.

Figure 49-38.GbE RMII Mode Signal Connection Block Diagram — External Clock Source

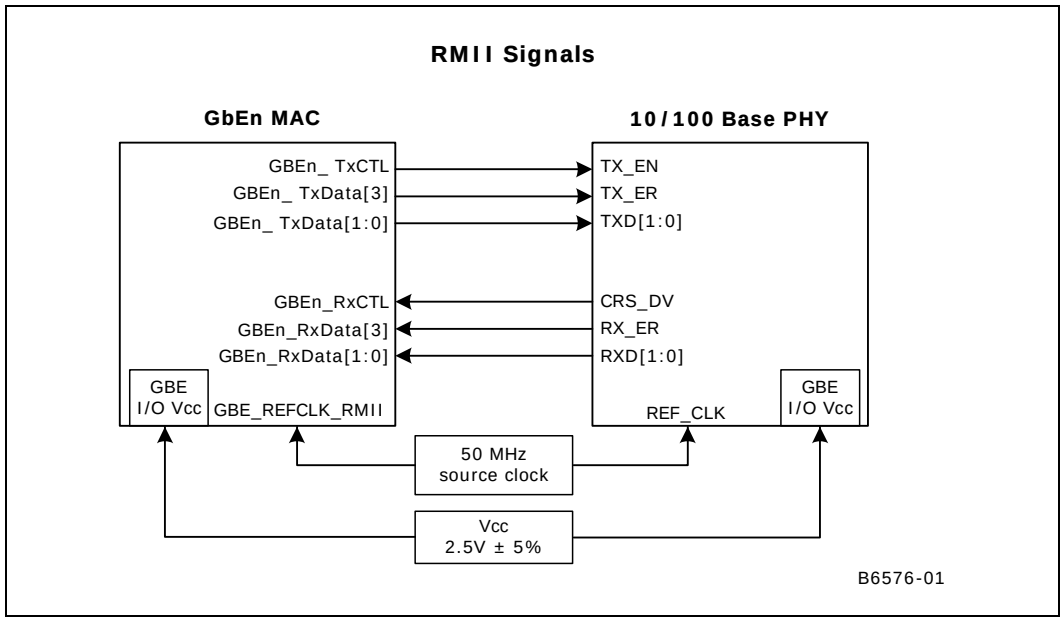


Figure 49-39 shows the timing diagram for RMII mode of operation.

Figure 49-39.GbE RMII Transmit and Receive Waveforms — RMII Mode

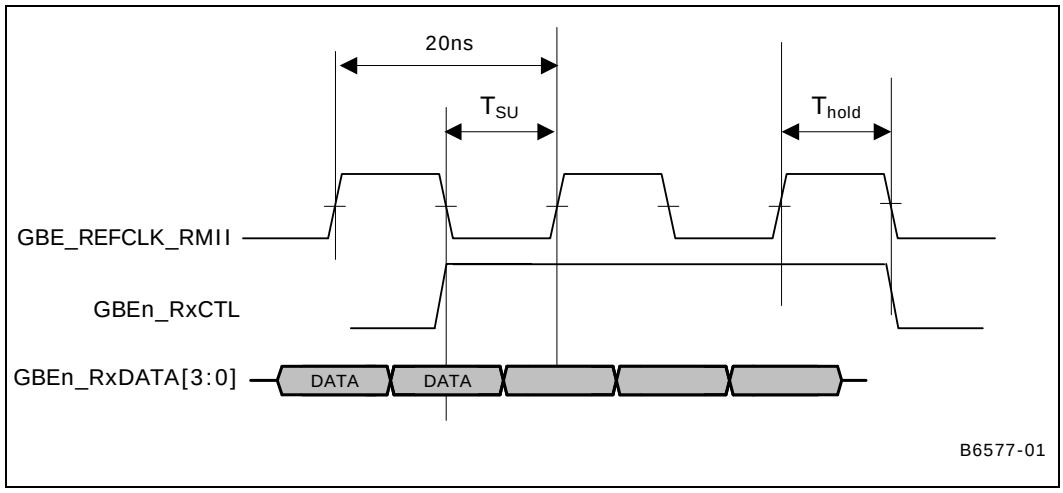


Table 49-79 shows the transmit and receive timing values for RMII 100/10 base mode of operation.

Table 49-79. GbE Transmit and Receive Timing Values — RMII 100/10 Base Mode

Symbol	Parameter	Min	Typical	Max	Units	Notes
	REF_CLK frequency	-	50	-	MHz	
	REF_CLK duty cycle	35		65	%	1
	REF_CLK jitter	-	-	80	ps	
Tsu	TXD[1:0], TX_EN,RXD[1:0], CRS_DV, RX_ER Data Setup to REF_CLK rising edge	4			ns	
Thold	TXD[1:0], TX_EN,RXD[1:0], CRS_DV, RX_ER Data hold from REF_CLK rising edge	2			ns	

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.13.3.8 MDIO Timing Specification

This section describes the AC timing for the MDIO interface.

Figure 49-40. MDIO Output Timing Diagram (EP80579 is Sourcing MDIO)

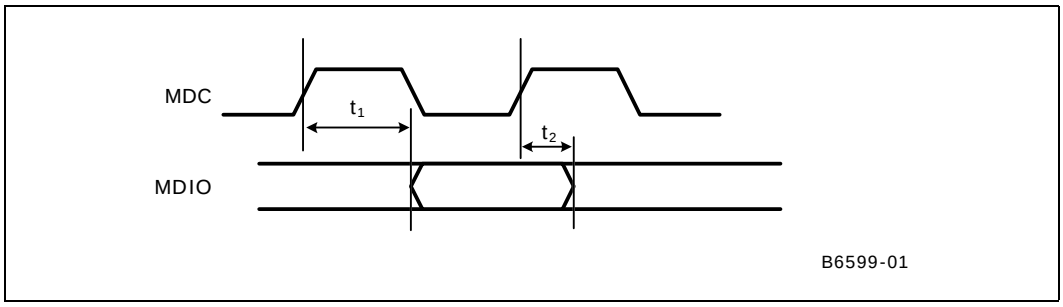


Figure 49-41. MDIO Input Timing Diagram (PHY is Sourcing MDIO)

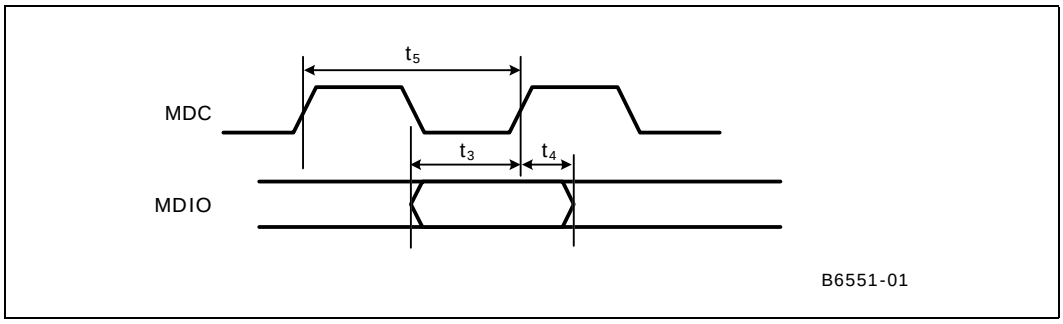


Table 49-80. MDIO Timings Values

Symbol	Parameter	Min	Max	Units	Figure	Notes
t_1	MDIO, clock to output timing with respect to rising edge of MDC clock		MDC/2 + 10 ns	ns	49-40	-
t_2	MDIO output hold timing after the rising edge of MDC clock	10		ns	49-40	-
t_3	MDIO input setup prior to rising edge of MDC clock	10		ns	49-41	-
t_4	MDIO input hold time after the rising edge of MDC clock	0	-	ns	49-41	-
t5	MDC clock period	-	500	ns	49-41	1

Notes:
1. The MDC clock period is 641 ns in SKU 2 and 8

49.5.13.3.9 EEPROM Timing Specification

This section describes the AC timing for the EEPROM interface.

Figure 49-42 shows a typical EEPROM read operation generated by the EP80579.

Figure 49-42. EEPROM Interface Timing Diagram

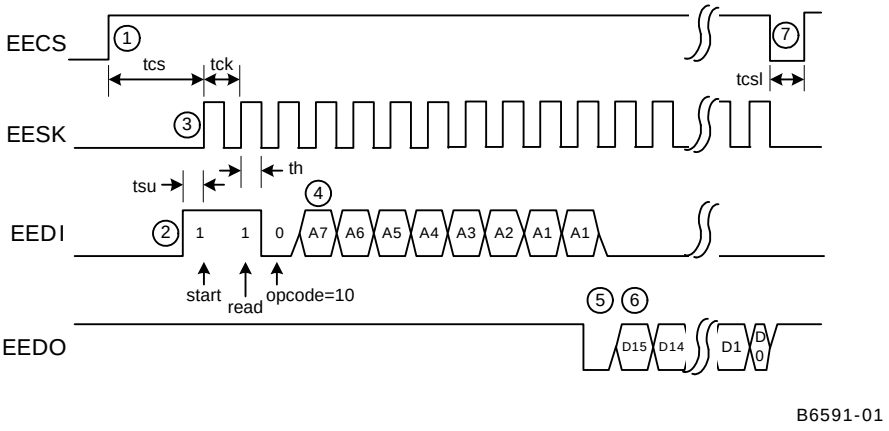


Table 49-81. EEPROM Read Operation (Sheet 1 of 2)

Step	Parameter
1	The EP80579 activates the EEPROM by asserting EECS.
2	After the chip select, The EP80579 starts driving data. It drives a 1 for the start bit, then a 10 (binary) for the read opcode.
3	The EP80579 starts driving the clock after driving the initial start bit. The EEPROM device latches a bit on every rising edge of the clock. The output data is driven near the falling edge of the clock to maximize setup and hold times. Likewise the read data is sampled near the falling edge of the clock.
4	After driving the start bit and read opcode EP80579 drives the address, starting at the most significant bit. A 256-word EEPROM requires 8 address bits (as shown in the diagram), while a 64-word EEPROM requires only 6 address bits.

Table 49-81. EEPROM Read Operation (Sheet 2 of 2)

Step	Parameter
5	An internal pullup is implemented on the EEDO pin so the pin initially floats high. When the EEPROM latches its last address bit it drives EEDO low. This can be used during the initial read to determine the size of the EEPROM. A 64-word EEPROM drives "data out" low after 6 address bits, while a 256-word EEPROM drives "data out" low after 8 address bits.
6	After driving its "data out" low, the EEPROM device drives 16 data bits, starting at the most significant bit. Each bit is driven after the rising edge of the clock.
7	After the read cycle, the EP80579 stops driving the clock and deasserts the chip select. When the chip select is deasserted the EEPROM stops driving the data.

Table 49-82. EEPROM Timing Values

Symbol	Parameter	Min	Max	Units	Figure	Notes
tcs	Chip select to first clock edge.	1024	3048	ns	49-42	1
tck	EEPROM clock cycle	1024	4160	ns	49-42	1
tsu	Setup time before rising clock edge.	480	-	ns	49-42	1
th	Hold time after rising clock edge.	480	-	ns	49-42	1
tcsl	Chip select deassertion time	992	-	ns	49-42	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.13.4 GbE Reset Conditions

There is no special reset sequence required for GbE I/O buffers. GbE receives a power-on reset signal from the CRU, but drives input data present on the input pins into the core during reset. Core logic forces all outputs to High-Z during reset.

49.5.13.5 RComp

The GbE RCOMP (resistive compensation) circuitry dynamically compensates the GbE I/O output drivers for variations in operating conditions due to process, temperature, voltage and PCB layout. These variations are measured through a resistive mechanism in two special I/O pads. The resistive mechanism on those I/O pads reference external resistors that the user provides to match output driver strength. Thus the output driver impedance can be tuned specifically to the application PCB characteristics for nominal signal transfer into the transmission lines formed by the PCB traces. The RCOMP design is nominally set to operate at 50 ohm, but the user is free to set the impedance in the range 45 ohm to 55 ohm.

Two RCOMP pins are provided to establish the GbE output driver impedance, one to control the drive high strength, and one to control the drive low strength. These RCOMP outputs drive into external resistors. The drivers form a resistor divider and the voltages developed in these dividers are compared to a reference voltage. The RCOMP state machine independently adjusts the strength of the drivers making them stronger or weaker until the comparator signals that the voltage is greater than or less than the reference. The state machine continues making adjustments causing the comparators to oscillate between two strength settings just above and just below the comparator trip point. Logic in the RCOMP state machine recognizes when this "dithering" between the two values has begun, and holds the strength output for the GbE outputs fixed at one of the two settings. Note this algorithm is independently and concurrently applied to the drive high strength and to the drive low strength.

The GbE RCOMP controller starts operation when the GBE_AUX_PWR_GOOD input is asserted. This condition indicates the power supplies are stable; the condition also indicates that the GBE_REFCLK input is being driven with a 125Mhz clock. The GBE_REFCLK input is divided by either 4 or 16 and then used to drive the RCOMP state machine. Software accessible registers that provide options to monitor/overwrite internal bias and comparator output are also present.

49.5.13.6 Voltage Domains

The GbE ports and associated miscellaneous I/Os (MDIO, EE, REFCLK, PME_WAKE, and RCOMP) are powered with 2.5V power supply. A secondary low current 3.3V power supply is also required to allow 3.3V input level tolerance. The GbE IO interface supports “wake-on-LAN” operation and therefore some of the I/O signals are powered with sustain power supplies that are always powered on while other I/O signals are powered with “core” power supplies that are only active when the remainder of the chip is powered on. GbE port 0 and the miscellaneous MDIO, EE, REFCLK, PME_WAKE, and RCMP IOs are powered through the sustain power supply VCCSUS25, and the 3.3V VCCGBEPSUS for 3.3V input tolerance. GbE ports 1 and 2 are powered with the VCC25 power supply and the VCCGBE33 is connected for 3.3V input level tolerance. The user may consider connecting VCCGBEPSUS power supply pins to the VCCPSUS power supply and connecting VCCGBE33 to the VCC33 power supply on the PCB in order to reduce the number of regulators used in the system. VCC25 and VCCSUS25 are only used in the GbE IO interface section of the chip.

49.5.14 Time Division Multiplex (TDM)

This section describes the electrical characteristics of the TDM interface.

49.5.14.1 TDM Signal List

For TDM pin descriptions, refer to Table 48-23, “TDM Interface Signals,” on page 1764.

49.5.14.2 TDM DC Characteristics

Table 49-83. TDM DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-84. TDM DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	1
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.14.3 TDM DC Clock Specification

Table 49-85. TDM DC Clock Input Specifications (RX_CLKn)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-86. TDM DC Clock Output Specifications (TX_CLKn)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.4	-		V	1
V_{OL}	Output voltage low	I_{out}=6mA		-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.14.4 TDM AC Characteristics

49.5.14.4.1 Transmit and Receive Timings

Table 49-87. TDM, Serial Timings Values

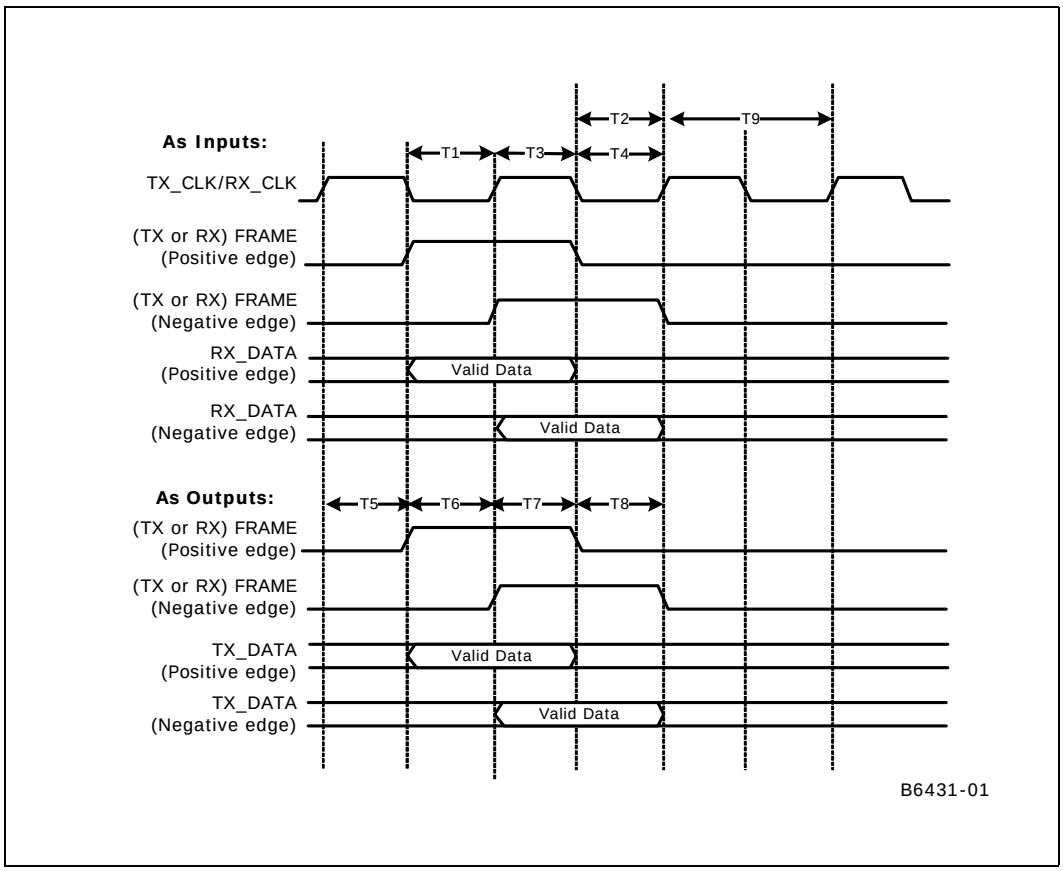
Symbol	Parameter	Min	Max	Units	Notes
T_1	Setup time of TX_FRAME, RX_FRAME, and RX_DATA_IN prior to the rising edge of clock	5		ns	1, 2, 3
T_2	Hold time of TX_FRAME, RX_FRAME, and RX_DATA_IN after the rising edge of clock	0		ns	1, 2, 3
T_3	Setup time of TX_FRAME, RX_FRAME, and RX_DATA_IN prior to the falling edge of clock	5		ns	1, 2, 3
T_4	Hold time of TX_FRAME, RX_FRAME, and RX_DATA_IN after the falling edge of clock	0		ns	1, 2, 3
T_5	Rising edge of clock to output delay for TX_FRAME, RX_FRAME, and TX_DATA_OUT		15	ns	1, 4
T_6	Falling edge of clock to output delay for TX_FRAME, RX_FRAME, and TX_DATA_OUT		15	ns	1, 3, 4
T_7	Output Hold Delay after rising edge of final clock for TX_FRAME, RX_FRAME, and TX_DATA_OUT	0		ns	1, 3, 4
T_8	Output Hold Delay after falling edge of final clock for TX_FRAME, RX_FRAME, and TX_DATA_OUT	0		ns	1, 3, 4
T_9	TX_CLK period and RX_CLK period	1/ 8.192 MHz	1/ 512 kHz	ns	5, 6

Notes:

1. TX_CLK and RX_CLK may be coming from external independent sources or being driven by the EP80579. The signals are shown to be synchronous for illustrative purposes and are not required to be synchronous.
2. Applicable to when the RX_FRAME and TX_FRAME signals are being driven by an external source as inputs into the EP80579. Always applicable to RX_DATA_IN.
3. The RX_FRAME and TX_FRAME can be configured to accept data on the rising or falling edge of the given reference clock. RX_FRAME and RX_DATA_IN signals are synchronous to RX_CLK and TX_FRAME and TX_DATA_OUT signals are synchronous to the TX_CLK.
4. Applicable to when the RX_FRAME and TX_FRAME signals are being driven by the EP80579 to an external source. Always applicable to TX_DATA_OUT.
5. The TX_CLK can be configured to be driven by an external source or be driven by the EP80579. The slowest clock speed that can be accepted or driven is 512 kHz. The maximum clock speed that can be accepted or driven is 8.192 MHz. The clock duty cycle accepted is 50/50 + 20%.
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.14.4.2 TDM Transmit and Receive Timing Diagrams

Figure 49-43.TDM, Serial Timings



49.5.15 Local Expansion Bus (LEB)

The Local Expansion bus is a 16-bit 33-80MHz bus with 8 programmable chip selects.

This section describes the electrical characteristics of the LEB interface.

49.5.15.1 LEB Signal List

For Local Expansion Bus pin descriptions, refer to Table 48-24, “Expansion Bus Signals” on page 1765.

49.5.15.2 LEB DC Characteristics

Table 49-88. LEB DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-89. LEB DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units
V_{OH}	Output voltage high	Iout=-6mA	2.0			V
V_{OL}	Output voltage low	Iout=6mA			0.4	V
I_{OH}	Output current at high voltage	V_{OH}=2.4	-20	-29	-35	mA
I_{OL}	Output current at low voltage	V_{OL}=0.8	20	21	24	mA

49.5.15.2.1 LEB DC Clock Specification

Table 49-90. LEB DC Clock Input Specifications (External Clock)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.15.3 LEB AC Characteristics

49.5.15.3.1 Local Expansion Bus Synchronous Operation

Figure 49-44. Local Expansion Bus Synchronous Timing

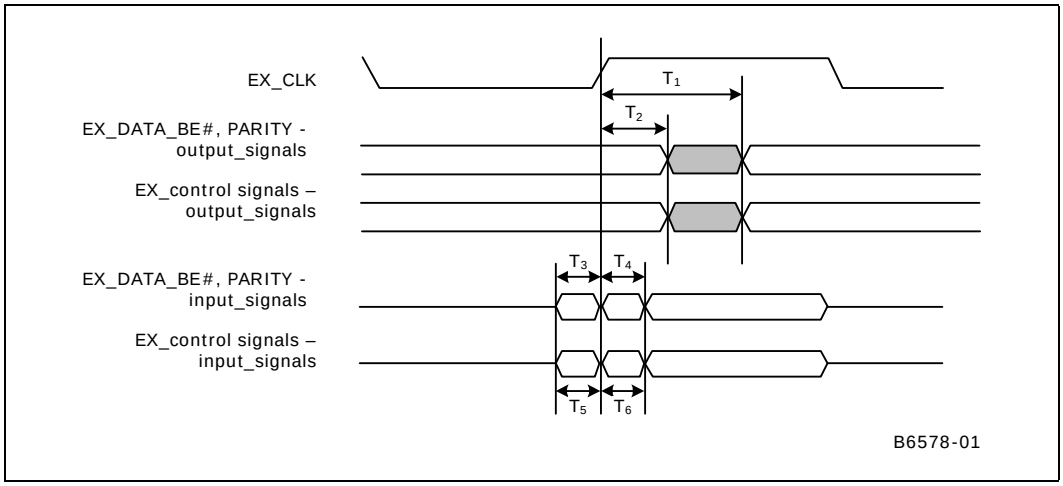


Table 49-91. Local Expansion Bus Synchronous Operation Timing Values

Symbol	Parameter	33 MHz		66 MHz		80 MHz		Units	Notes
		Min	Max	Min	Max	Min	Max		
T_1	Valid rising edge of EX_CLK to valid signal on the output.		8.5		8.0		7.5	ns	1
T_2	Valid signal hold time after the rising edge of EX_CLK	1		1		1		ns	1, 4
T_3	Valid data signal on an input prior to the rising edge of EX_CLK	2.5		2.5		2.5		ns	1
T_4	Required hold time of a data input after the rising edge of EX_CLK	0.5		0.5		0.5		ns	1, 4
T_5	Valid control signal on an input prior to the rising edge of EX_CLK	3.5		3.5		3.5		ns	1
T_6	Required hold time of a control input after the rising edge of EX_CLK	0.5		0.5		0.5		ns	1, 4
CLoad	Load Capacitance	5	60	5	50	5	40	pF	4

Notes:
1. Drive settings do not apply to EX_CS# signals and are expected to be point to point.
2. EX_control_signals output signals consist of EX_ALE, EX_ADDR, EX_CS#, EX_RD#, EX_WR#, EX_WAIT#
3. EX_control_signals input signals consist of EX_ADDR, EX_CS#, EX_BURST, EX_RD#, EX_WR#
4. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.16 Controller Area Network (CAN)

The EP80579 provides a CAN interface that includes support for CAN 2.0B protocol, 11-bit and 29-bit identifiers, and supports bit rates up to 1 Mbps. The EP80579 provides support for various external PHY chips, such as the TI-SN65HVD230D transceiver.

This section describes the electrical characteristics of the CAN interface.

49.5.16.1 CAN Signal List

For a pin description for the CAN Interface, refer to Table 48-21, “Controller Area Network Bus Signals” on page 1759.

49.5.16.2 CAN DC Characteristics

Table 49-92. CAN DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:

1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-93. CAN DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V
I_{OH}	Output current at high voltage	V_{out}=2.4	-12	-27	-48	mA
I_{OL}	Output current at low voltage	V_{out}=0.8	12	18	26	mA

49.5.17 Synchronous Serial Port (SSP)

This section describes the electrical characteristics of the SSP interface.

49.5.17.1 SSP Signal List

For a pin description for the SSP Interface, refer to Table 48-25, “SSP Interface Signals” on page 1768.

49.5.17.2 SSP DC Characteristics

This section describes the DC characteristics of the SSP interface.

Table 49-94. SSP DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.17.2.1 SSP DC Output Characteristics

Table 49-95. SSP DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	1
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

Table 49-96. SSP DC Clock Specification

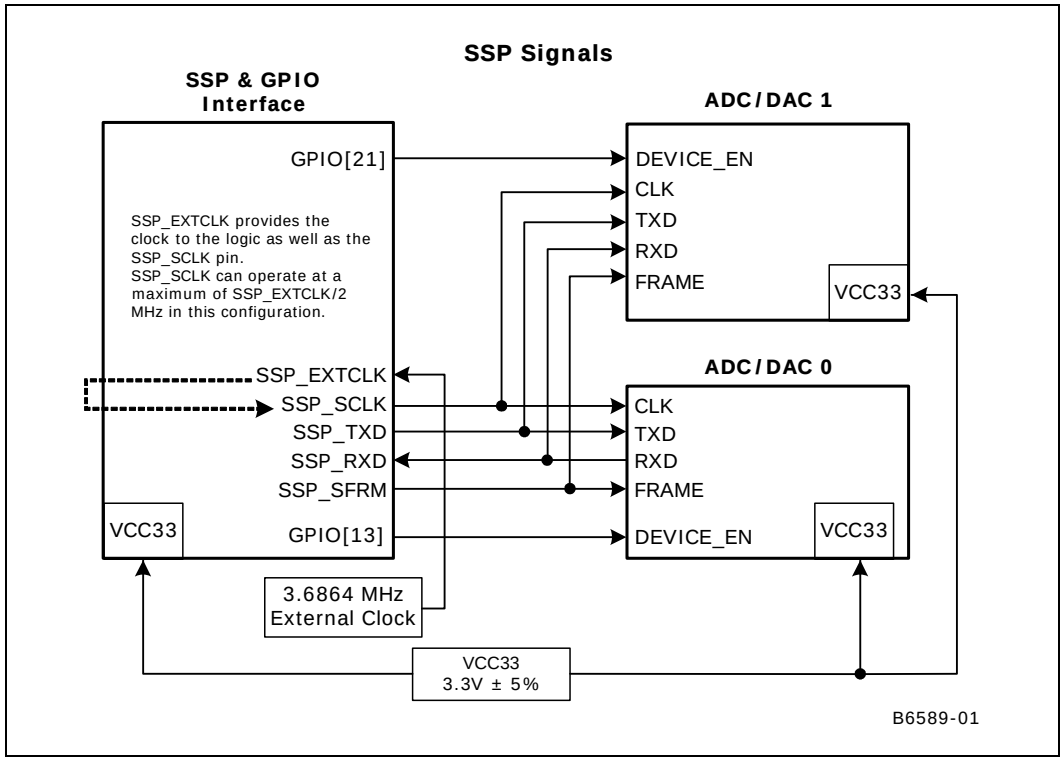
Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-		V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{OL}	Output current at low voltage	V_{OL}<0.8	-	-	4	mA	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
Cin	Input pin cap	-	-	-	5	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.17.2.2 SSP Block Diagram

Figure 49-45 shows the SSP signal connections used when connecting to multiple analog-to-digital convertors, with a multi-bus connection, and using an external clock generation mode.

Figure 49-45.SSP Signal Connection Block Diagram - Multi-Drop Connections



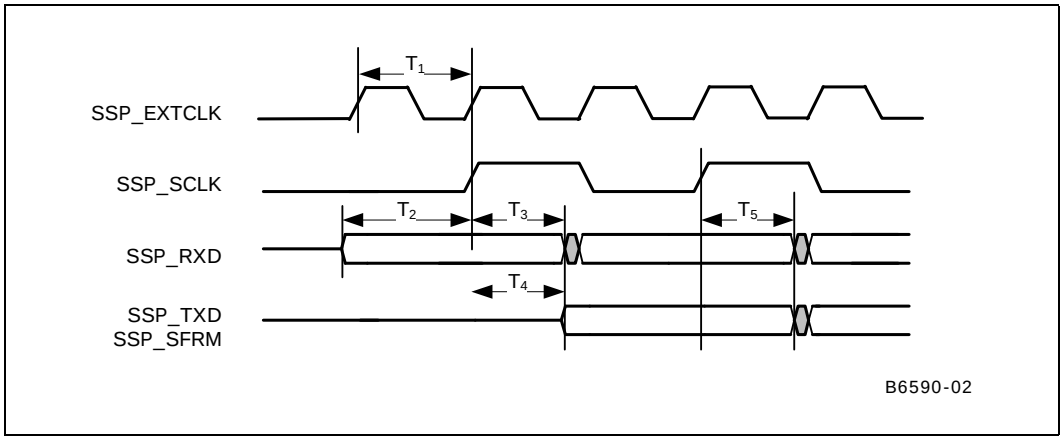
49.5.17.3 SSP AC Characteristics

Table 49-97. SSP Timing Values and Test Conditions

Symbol	Parameter	Min	Nominal	Max	Units	Notes
SSP_SCLK	Clock period of SSP_SCLK when the clock is being generated from the internally generated 3.6864 MHz clock	-	271	-	ns	1, 3, 4
-	SSP_SCLK clock period that can be produced when the clock is being generated from the external 3.6864 MHz clock via SSP_EXTCLK	542	-	-	ns	1, 3
SSP_EXTCLK	Clock period when the clock is being generated from the externally supplied maximum clock rate of 3.6864 MHz clock	271	-	-	ns	1, 3
T_1	Output Valid Delay from SSP_EXTCLK to SSP_SCLK in external clock mode	2	-	15	ns	1, 3
T_2	Input Setup time for data prior to the valid edge of SSP_SCLK. These signals include SSP_RXD.	15	-	-	ns	1, 5
T_3	Input hold time for data after the valid edge of SSP_SCLK. These signals include SSP_SRXD.	0	-	-	ns	1, 3, 5
T_4	SSP_SCLK clock to output valid delay from output signals. These signals include SSP_TXD and SSP_SFRM.	-	-	6	ns	1, 2, 3, 5
T_5	Output data hold valid from valid edge of SSP_SCLK. These signals include SSP_TXD and SSP_SFRM.	1	-	-	ns	1, 2, 3, 5

Notes:
1. Timing was designed for a system load between 5pF and 40pF
2. Clock jitter on the SSP_SCLK is designed to be an average of the specified clock frequency. The SSPSCLK jitter specification is unspecified.
3. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.
4. For low-power SKU configured for internal mode the SSP_SCLK is 360 ns from an internal generated 2.777 MHz clock.
5. For reference purposes, the timing diagram shows a positive clock edge launch and a positive clock edge capture, however, depending on the frame format selected, each transmitted bit is driven on either the rising or falling edge of SSP_SCLK, and is sampled on the opposite clock edge.

Figure 49-46. SSP Interface Timing Diagram



49.5.18 IEEE 1588-2008 Hardware Assist Interface

This section describes the electrical characteristics of the IEEE 1588-2008 Hardware Assist Interface.

49.5.18.1 IEEE 1588-2008 Hardware Assist Signal List

For a pin description for the IEEE 1588-2008 Hardware Assist Interface, refer to Table 48-26, “IEEE 1588-2008 Hardware Assist Interface Signals” on page 1768.

49.5.18.2 IEEE 1588-2008 Hardware Assist DC Characteristics

Table 49-98. IEEE 1588-2008 Hardware Assist DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-99. IEEE 1588-2008 Hardware Assist DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	1
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.18.3 IEEE 1588-2008 Hardware Assist AC Characteristics

The IEEE 1588-2008 Hardware Assist Interface is an edge-dependent interface and has no critical AC timing requirements.

49.5.19 IICH Miscellaneous Signals (PME#, PCIRST#, SPKR)

This section describes the electrical characteristics of the IICH Miscellaneous signals.

49.5.19.1 IICH Miscellaneous Signal List

For a pin description for the IICH Miscellaneous signals, refer to Table 48-28, “Miscellaneous Signals” on page 1770.

49.5.19.2 IICH Miscellaneous Signals DC Characteristics

Table 49-100.IICH Miscellaneous Signals DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-101.IICH Miscellaneous Signals DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-6mA	2.0	-	-	V	1
V_{OL}	Output voltage low	I_{out}=6mA	-	-	0.4	V	-
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA	-
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA	-

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.19.3 IICH Miscellaneous Signals AC Characteristics

The AC electrical specifications of the EP80579 IICH Miscellaneous signals are compliant with the *PCI Local Bus Specification*, Rev. 3.0 (refer to AC Specifications for 3.3V Signaling for more information).

49.5.20 Clock Resource Unit (CRU)

The CRU unit is responsible for generating and distributing the clocks to the various functional units within the SOC.

This section describes the electrical characteristics of the Global Clock signals.

49.5.20.1 CRU Signal List

For a pin description for the CRU Signals, refer to Table 48-5, “Global Clock and Reset (CRU) Signals” on page 1736.

49.5.20.2 CRU DC Characteristics

Table 49-102.CRU Differential Clock DC Specifications

Symbol	Parameter	Min	Typical	Max	Units	Figure	Notes
V_{IL}	Input Low Voltage	-0.150	0.00	0.150	V	49-47	1
V_{IH}	Input High Voltage	0.660	0.710	0.850	V	49-47	1
$V_{CROSS(abs)}$	Absolute Crossing Point	0.250	N/A	0.550	V	49-47, 49-48	1, 2, 3, 6, 9
$V_{CROSS(rel)}$	Relative Crossing Point	0.250+0.5(V_{HAVg}-0.71)	N/A	0.550+0.5(V_{HAVg}-0.71)	V	49-47, 49-48	1, 2, 3, 6, 7, 9
ΔV_{CROSS}	Range of Crossing Point	N/A	N/A	0.140	V	49-47, 49-48	1, 2, 8, 9
V_{RBM}	Ringback Margin	0.200	N/A	N/A	V	49-47	1, 4, 9
V_{TM}	Threshold Margin	V_{CROSS}-0.10	N/A	V_{CROSS}+0.10	V	49-47	1, 5, 9

Notes:
1. Unless otherwise noted, all specifications in this table apply to all processor frequencies.
2. Crossing voltage is defined as the instantaneous voltage value when the rising edge of CLKP100 equals the falling edge of CLKN100.
3. VHavg is the statistical average of the VH measured by the oscilloscope.
4. Ringback Margin is defined as the absolute voltage difference between the maximum Rising Edge Ringback and the maximum Falling Edge Ringback.
5. Threshold Region is defined as a region entered around the crossing point voltage in which the differential receiver switches. It includes input threshold hysteresis.
6. The crossing point must meet the absolute and relative crossing point specifications simultaneously.
7. VHavg can be measured directly using "Vtop" on Agilent* scopes and "High" on Tektronix* scopes.
8. ΔV_{CROSS} is defined as the total variation of all crossing voltages as defined in note 2.
9. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Figure 49-47.CRU Differential Clock Waveform

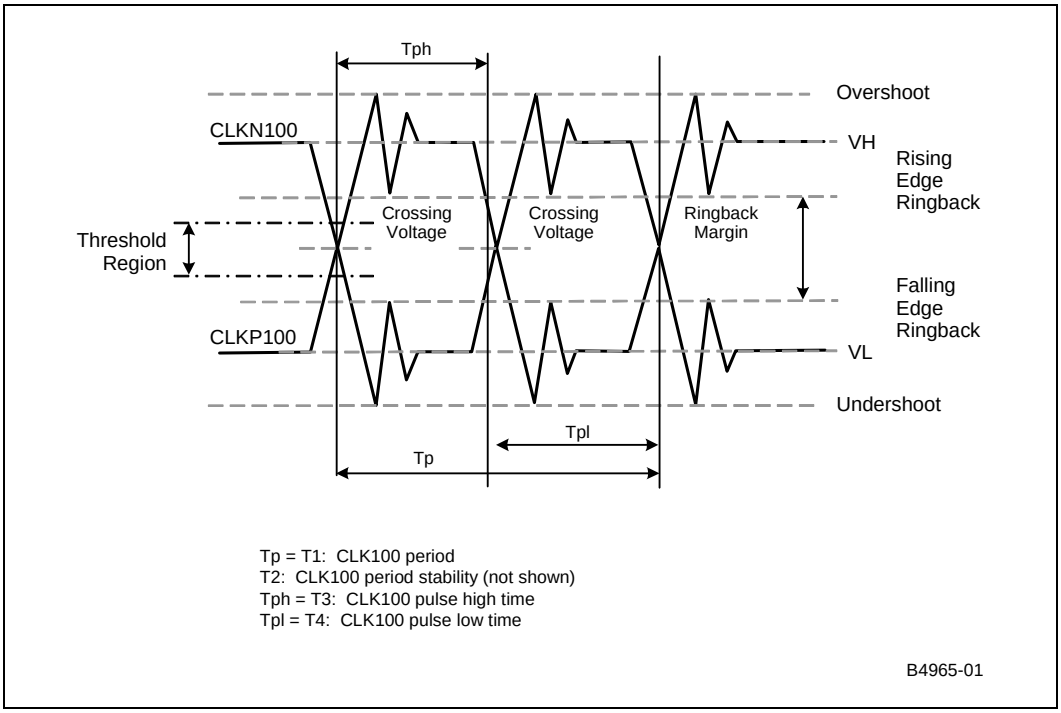
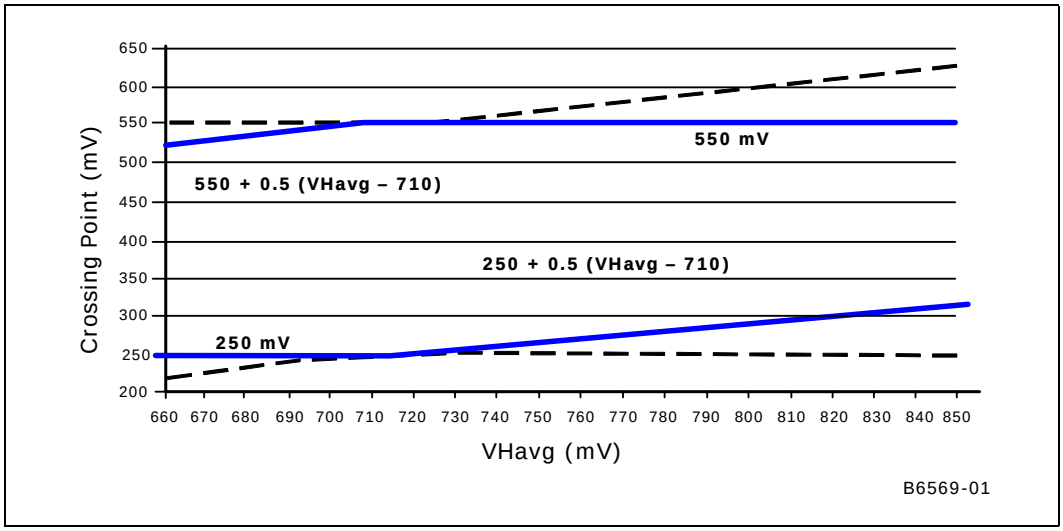


Figure 49-48.CRU Differential Clock Cross-Point Specification



49.5.20.3 CRU AC Specifications

Table 49-103.CRU Differential Input Clock Timing Specifications (Sheet 1 of 2)

Symbol	Parameter	Min	Nom	Max	Unit	Figures	Notes
CLKN100/CLKP100 (100MHz FSB clock)							
-	CLK100 Frequency		-	100	MHz	-	-
-	Duty Cycle	45	-	55	%	-	-
T1	CLK100 Period	9.997	-	10.20	ns	49-47	1, 6
T2	CLK100 Period Stability	N/A	-	200.00	ps	-	2, 6
-	Frequency Tolerance	-	-	350	ppm	-	-
-	Clock Jitter	-	50	150	ps	-	-
T3	T_{PH} CLK100 Pulse High Time	3.94	5.00	6.12	ns	49-47	4, 6
T4	T_{PL} CLK100 Pulse Low Time	3.94	5.00	6.12	ns	49-47	4, 6
T5	CLK100 Rise Time	175	-	700	ps	49-47	3, 5, 6
T6	CLK100 Fall Time	175	-	700	ps	49-47	3, 5, 6

Notes:

1. The period specified here is the average period. A given period may vary from this specification as governed by the period Stability specification (T2).
2. In this context, period stability is defined as the worst case timing difference between successive crossover voltages. In other words, the largest absolute difference between adjacent clock periods must be less than the period stability.
3. Slew rate is measured between the 35% and 65% points of the clock swing (VL to VH).
4. Combining the longest clock-high and clock-low times would violate the max clock period, and that combining the shortest clock-high and clock-low times would violate the minimum clock period. The clock-high and clock-low times specify the most extreme allowable combination of clock period and duty cycle.
5. Slew rate specifications apply to both rising and falling edges.
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-103.CRU Differential Input Clock Timing Specifications (Sheet 2 of 2)

Symbol	Parameter	Min	Nom	Max	Unit	Figures	Notes
CLKN100/CLKP100 (133MHz FSB clock)							
-	CLK100 Frequency		-	133	MHz	-	-
-	Duty Cycle	45	-	55	%	-	6
T1	CLK100 Period	7.497	-	7.65	ns	49-47	1, 6
T2	CLK100 Period Stability	N/A	-	200	ps		2, 6
-	Frequency Tolerance	-	-	350	ppm	-	-
-	Clock Jitter	-	50	150	ps	-	-
T3	T_{PH} CLK100 Pulse High Time	3.375	3.75	4.2075	ns	49-47	4, 6
T4	T_{PL} CLK100 Pulse Low Time	3.375	3.75	4.2075	ns	49-47	4, 6
T5	CLK100 Rise Time	175		700	ps	49-47	3, 5, 6
T6	CLK100 Fall Time	175		700	ps	49-47	3, 5, 6

Notes:
1. The period specified here is the average period. A given period may vary from this specification as governed by the period Stability specification (T2).
2. In this context, period stability is defined as the worst case timing difference between successive crossover voltages. In other words, the largest absolute difference between adjacent clock periods must be less than the period stability.
3. Slew rate is measured between the 35% and 65% points of the clock swing (VL to VH).
4. Combining the longest clock-high and clock-low times would violate the max clock period, and that combining the shortest clock-high and clock-low times would violate the minimum clock period. The clock-high and clock-low times specify the most extreme allowable combination of clock period and duty cycle.
5. Slew rate specifications apply to both rising and falling edges.
6. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.21 Sideband Miscellaneous Signals

This section describes the electrical characteristics of the Sideband Miscellaneous signals.

49.5.21.1 Sideband Miscellaneous Signals Signal List

For a pin description for these sideband signals, refer to Table 48-6, “Sideband Miscellaneous Signals” on page 1737.

49.5.21.2 Sideband Miscellaneous Signals DC Characteristics

Table 49-104.Sideband Miscellaneous Signals DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	$0<V_{IN}<Vcc3\_3$	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

Table 49-105.Sideband Miscellaneous Signals DC Output Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{OH}	Output voltage high	I_{out}=-0.5mA	0.9VCC33	-	-	V	1
V_{OL}	Output voltage low	I_{out}=1.5mA	-	-	0.1VCC33	V	
I_{OH}	Output current at high voltage	V_{OH}>0.9VCC33	-0.5	-	-	mA	
I_{OL}	Output current at low voltage	V_{OL}<0.1VCC33	-	-	1.5	mA	

Notes:
1. VOH spec does not apply to signals that are open drain driver. Open drain signals must have an external pull up resistor.

49.5.21.3 Sideband Miscellaneous Signals AC Characteristics

Refer to Figure 6-6, “Powergood Reset Sequence” on page 172 and Figure 6-7, “Hard Reset Sequence” on page 173 for Sideband Miscellaneous Signal timing details.

49.5.22 IMCH Reset

This section describes the electrical characteristics of the IMCH Reset signals.

49.5.22.1 IMCH Reset Signal List

For a pin description for the IMCH Reset Signals, refer to Table 48-7, “IMCH Reset Signals” on page 1739.

49.5.22.2 IMCH Reset Signals DC Characteristics

Table 49-106.IMCH Reset Signals DC Input Characteristics

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	Input voltage high	-	2.0	-	-	V	-
V_{IL}	Input voltage low	-	-	-	0.8	V	-
I_{leak}	Input Leakage Current	0<V_{IN}<Vcc3_3	-	-	10	µA	-
C_{IN}	Input-pin capacitance	-	-	-	10	pf	1

Notes:
1. Guaranteed by design. These values are typical values seen for this process, but not measured during production testing.

49.5.22.3 IMCH Reset Signals AC Characteristics

Refer to Figure 6-6, “Powergood Reset Sequence” on page 172 and Figure 6-7, “Hard Reset Sequence” on page 173 for IMCH Reset Signal timing details.

49.5.23 JTAG

This section describes the electrical characteristics of the JTAG interface.

49.5.23.1 JTAG Signal List

For a pin description for the JTAG Signals, refer to Table 48-27, "JTAG Interface Signals" on page 1769.

49.5.23.2 JTAG DC Characteristics

Table 49-107.JTAG DC Specifications (except BPM4_PRDY_OUT)

Symbol	Parameter	Conditions	Min	Typical	Max	Units	Notes
V_{IH}	TMS, TDI, TRST# Input-high voltage		0.8		1.4	V	2
V_{IL}	TMS, TDI, TRST# Input-low voltage		-0.5		0.4	V	2
R_{ON}	TDO Ron impedance		5		16	Ω	3

Notes:
1. 1.2V I/O buffers
2. These values are typical values seen by the manufacturing process and are not tested
3. Guaranteed by design.

Table 49-108.JTAG DC Output Specifications (BPM4_PRDY_OUT)

Symbol	Parameter	Conditions	Min	Typical	Max	Units
V_{OH}	Output voltage high	I_{out}=-6mA	2.0			V
V_{OL}	Output voltage low	I_{out}=6mA			0.4	V
I_{OH}	Output current at high voltage	V_{OH}=2.4	-12	-27	-48	mA
I_{OL}	Output current at low voltage	V_{OL}=0.8	12	18	26	mA

Note: 3.3 V I/O buffers

49.5.23.3 JTAG AC Characteristics

Table 49-109.JTAG Timing Specifications (Sheet 1 of 2)

Symbol	Parameter	Conditions	Min	Max	Units	Figures	Notes
T_{JTF}	TCK Frequency	TCK at 25 MHz and period 40ns	0	25	MHz	-	
T_{TCH}	TCK High Time	Measured at 0.6V	7.0		ns	-	1, 5
T_{TCL}	TCK Low Time	Measured at 0.6V	7.0		ns	-	1, 5

Notes:
1. 1.2V I/O buffers
2. These values are typical values seen by the manufacturing process and are not tested
3. For TMS, TDI and TRST#.
4. TCK input threshold voltage.
5. Guaranteed by design.

Table 49-109.JTAG Timing Specifications (Sheet 2 of 2)

Symbol	Parameter	Conditions	Min	Max	Units	Figures	Notes
T_{TCR}	TCK Rise Time	0.3V to 0.72V		5	ns	-	1, 5
T_{TCF}	TCK Fall Time	0.72V to 0.3V		5	ns	-	1, 5
T_{TIS1}	Input Setup to TCK--TDI, TMS		3.0		ns	49-50	3
T_{TIH1}	Input Hold from TCK--TDI, TMS		2.0		ns	49-50	3
T_{TOV1}	TDO Output Valid Delay	Relative to falling edge of TCK	4.25	13.25	ns	49-49	2, 5
T_{OF1}	TDO Float Delay	Relative to falling edge of TCK	4.25	13.25	ns	49-49	4, 5

Notes:
1. 1.2V I/O buffers
2. These values are typical values seen by the manufacturing process and are not tested
3. For TMS, TDI and TRST#.
4. TCK input threshold voltage.
5. Guaranteed by design.

49.5.23.3.1 JTAG Timing Diagrams and Test Conditions

Figure 49-49.JTAG Output Timing Measurement Waveforms

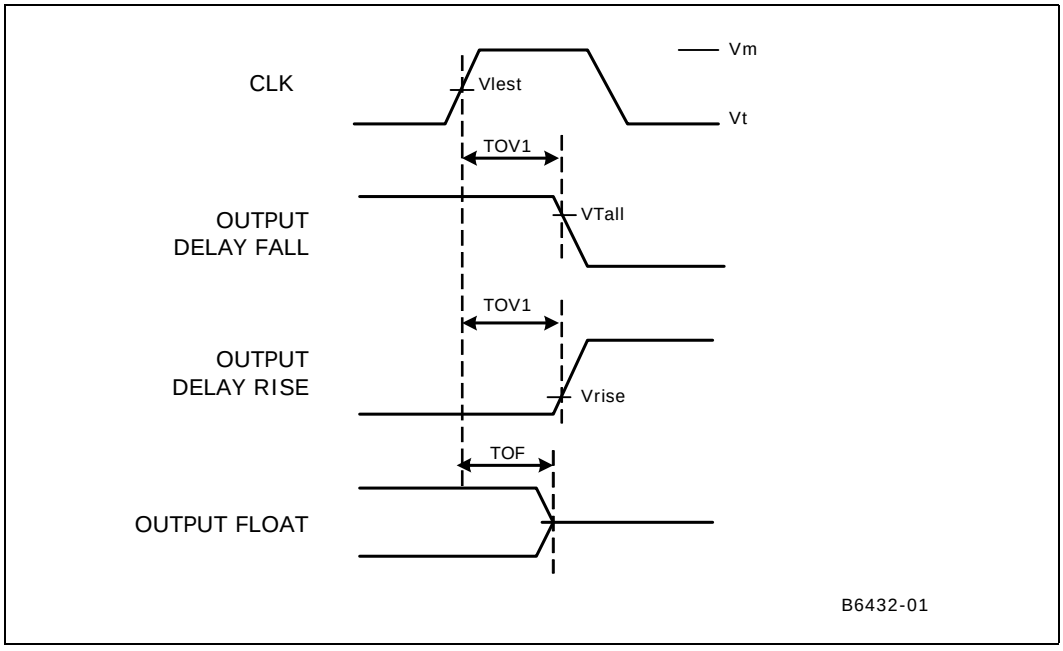
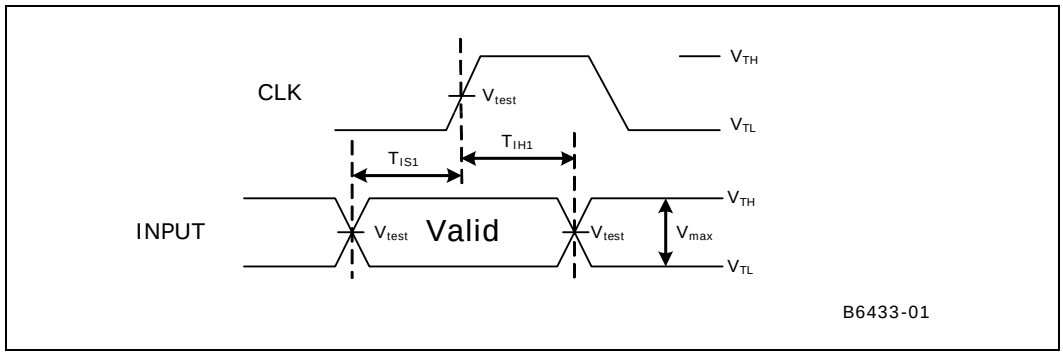


Figure 49-50.JTAG Input Timing Measurement Waveforms



§ §

50.0 Thermal Specifications and Design Considerations

The EP80579 requires a thermal solution that maintains the maximum case temperature defined in Table 50-1, “EP80579 Thermal Design Power (TDP) and Maximum Case Temperature Specifications (TC-MAX)” . Any attempt to operate outside these operating limits may result in permanent damage to the processor and potentially other components in the system. As processor technology changes, thermal management becomes increasingly crucial when building computer systems. Maintaining the proper thermal environment is key to reliable long-term system operation. A complete thermal solution includes both component-level and system-level thermal management features. Refer to the *Intel® EP80579 Integrated Processor product line Thermal Design Guide* for a thorough treatment of thermal design considerations.

To allow for the optimal operation and long-term reliability of Intel® architecture-based systems, the system/processor thermal solution must be designed such that the processor remains within the minimum and maximum case temperature ($T_{C\text{-}MAX}$) specifications at the corresponding Thermal Design Power (TDP) value listed in Table 50-1. Thermal solutions that are not designed to provide this level of thermal capability may affect the long-term reliability of the processor and the system. The case temperature is defined at the geometric top center of the Integrated Heat Spreader (IHS).

The EP80579 monitors die temperature using athermal sensor.The thermal sensor (described in Section 50.2, “Thermal Sensor”) must be used to determine when the maximum specified component temperature has been reached. The on-die thermal sensor is designed to help protect the processor in the unlikely event that an application exceeds the TDP recommendation for a sustained period of time. For more details on the usage of this feature, refer to Section 50.2, “Thermal Sensor”. In all cases, the thermal sensor feature must be enabled for the processor to remain within the operating limits.

50.1 Thermal Characteristics

Consult the *Intel® EP80579 Integrated Processor product line Thermal Design Guide* for complete information about thermal characteristics. Contact your local Intel sales office or your distributor to obtain the latest version.

50.1.1 Specifications

50.1.2 Thermal Design Power (TDP) Dissipation

The Thermal Design Power (TDP) values in Table 50-1 assume full-speed operation by all peripherals and internal components in a realistic application. This TDP value provides a high-level estimate for target applications.

The processor's power is specified as Thermal Design Power (TDP) for thermal solution design. TDP is defined as a system design target associated with the maximum component operating temperature specifications. TDP values are determined based on typical DC electrical specification and maximum component temperature for a realistic-case application running at maximum utilization. The Intel TDP specification is a recommended design point and is not representative of the absolute maximum power the processor may dissipate under worst-case conditions. For any excursions beyond TDP, the processor passive cooling feature is available to maintain the processor thermal specifications. Refer to Section 50.2, "Thermal Sensor"

Table 50-1. EP80579 Thermal Design Power (TDP) and Maximum Case Temperature Specifications ($T_{C\text{-}MAX}$)

	Intel® EP80579 Integrated Processor				Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology			
SKU	**2**	**8**	**4**	**6**	**1**	**3**	**7**	**5**
Processor Frequency (MHz)	600	600	1066	1200	600	1066	1066	1200
TDP[1] (Watts)	11	11	18	19	13	20	20	21
$T_{CASE\text{-}MAX}$	100°C	100°C	97°C	95°C	100°C	97°C	97°C	95°C

Notes:
1. Thermal Design Power (TDP) is a system design target associated with the maximum component operating temperature specifications. TDP values are determined based on typical DC electrical specification and maximum component temperature for a realistic-case application running at maximum utilization.

Refer to Section 50.0, "Thermal Specifications and Design Considerations" for further thermal specifications.

50.2 Thermal Sensor

An independent thermal sensor helps control the silicon temperature by activating the thermal control sequence when the EP80579 reaches its maximum operating temperature. The temperature at which the EP80579 activates this thermal control sequence is not user configurable and is not visible to the software.

The thermal sensor is an independent sensor that provides two outputs, THRMTRIP# and PROCHOT#. Refer to Section 50.2.1, "Catastrophic Thermal Protection" for details about THRMTRIP# and to Section 50.2.2, "Thermal Sensor Features" for details about PROCHOT#.

The EP80579 provides the IA-32 core thermal monitor. The IA-32 core thermal monitor has a throttling capability that can be activated via software or pre-boot firmware. Refer to Section 50.2.2.2, "Processor Passive Cooling" and Section 50.2.2.3, "On-Demand Passive Cooling" for details.

50.2.1 Catastrophic Thermal Protection

The EP80579 provides catastrophic thermal protection via an on-die thermal sensor that detects when the silicon has reached approximately 20°C above $T_{CASE-MAX}$. This catastrophic trip point is programmed at the factory. The catastrophic trip point will initiate the Processor Thermal Trip control sequence, which is described in Section 50.2.1.1, "THRMTRIP# Control Sequence". Once the thermal sensor has detected this over heat condition, THRMTRIP# is asserted.

THRMTRIP# is also available as an input pin to support platform thermal management requirements. An external thermal sensor could be implemented to monitor the platform thermal condition, asserting THRMTRIP# when catastrophic thermal conditions exist on the platform. The EP80579 receives this signal and initiates the THRMTRIP# Control Sequence described in Section 50.2.1.1, "THRMTRIP# Control Sequence".

Note: Intel recommends that an external thermal sensor be used to protect the EP80579 and the system against excessive temperature. Even with the activation of THRMTRIP#, which halts all internal clocks and activity, leakage current can be high enough such that the EP80579 cannot be protected in all conditions without completely removing the supplied power source. If the external thermal sensor detects the silicon has reached a catastrophic temperature approximately 20°C above TCASE-MAX, or if the THRMTRIP# signal is asserted, the VCC supply to the processor must be turned off within 500 ms to prevent silicon damage due to thermal runaway.

50.2.1.1 THRMTRIP# Control Sequence

A catastrophic thermal trip event will assert THRMTRIP#, which indicates that the on-die thermal sensor has detected an overheat condition. Intel recommends that immediate action be taken to prevent silicon damage.

The temperature which the EP80579 activates this thermal control sequence is not user configurable and is not software visible. The following list provides additional information:

- If THRMTRIP# goes active, the processor is indicating an overheat condition, and will immediately transition to an S5 state. However, since the processor has overheated, it will not respond to the STPCLK# pin with a stop grant special cycle. Therefore, the EP80579 will not wait for one.
- Immediately upon seeing THRMTRIP# low, EP80579 will initiate a transition to the S5 state, drive signals SLP_S3#, SLP_S4#, SLP_S5# low, and set the CTS bit. The transition will generally look like a power button override.
- When a THRMTRIP# event occurs, the EP80579 must power down immediately without following the normal S0 -> S5 path. This can happen in parallel, but the EP80579 must immediately enter a power down state. The EP80579 will do this by driving signals SLP_S3#, SLP_S4#, and SLP_S5# within 3 PCICLKs after sampling THRMTRIP# active.
- If the processor is running extremely hot and is heating up, it is possible (although very unlikely) that components around it are no longer executing cycles properly. Therefore, if THRMTRIP# goes active, any other components that are relying on state machine logic to perform the power down, may not power down correctly. Because the state machine may not be working, the system may not power down completely.

The EP80579 will follow the flow in the steps that follow for THRMTRIP#:

1. At boot (PLTRST# low), THRMTRIP# ignored.
2. After power-up (PLTRST# high), if THRMTRIP# sampled active, SLP_S3#, SLP_S4#, and SLP_S5# fire, and normal sequence of sleep machine starts.

3. Until sleep machine enters the S5 state, SLP_S3#, SLP_S4#, and SLP_S5# stay active, even if THRMTRIP# is now inactive. This is the equivalent of “latching” the thermal trip event.
4. When the S5 state reached, return to step 1 above, otherwise stay here. If the EP80579 never gets to S5, the EP80579 does not reboot until power is cycled.

During boot, THRMTRIP# is ignored until SLP_S3#, PWROK, VRMPWRGD, and PLTRST# are all ‘1’. During entry into a powered-down state (due to S3,S4, S5 entry, power cycle reset, etc.) THRMTRIP# is ignored until any of the following occurs:

- SLP_S3# = 0
- PWROK = 0
- VRMPWRGD = 0

1. Set the AFTERG3_EN bit
2. Clear the PWRBTN_STS bit
3. Clear all the GPE0_EN register bits
4. Clear the SMB_WAK_STS bit only if SMB_WAK_STS was set due to SMBus slave receiving message and not set due to SMBAlert.

Note: The THRMTRIP# pin must be glitch free.

50.2.2 Thermal Sensor Features

PROCHOT# is asserted when the silicon temperature has reached the maximum operating limit. If the EP80579 reaches the trip temperature indicated in Table 50-1, “EP80579 Thermal Design Power (TDP) and Maximum Case Temperature Specifications (TC-MAX)” , then PROCHOT# is asserted and the EP80579 initiates a SMI# or SCI.

PROCHOT# provides platform visibility of a thermal condition determined by the thermal sensor. PROCHOT# should be implemented in the platform design as a thermal warning, by initiating passive cooling, turning on fans or power management.

When PROCHOT# is asserted, Intel recommends that power management software begins a processor-initiated passive cooling routine, which is described in Section 50.2.2.2, “Processor Passive Cooling”.

For platform thermal sensor designs, on-demand throttling is provided. Section 50.2.2.3, “On-Demand Passive Cooling” provides more information.

50.2.2.1 PROCHOT# Control Sequence

The PROCHOT# signal is a status output pin from the thermal sensor. The signal indicates that the EP80579 has reached the maximum operating limit. When this maximum operating temperature is reached, power management software or pre-boot firmware should take action to reduce the temperature. The EP80579 follows these behaviors with regard to the PROCHOT# signal:

- Based on the PROCHOT# signal activation, the EP80579 generates an SMI# or SCI (depending on SCI_EN).
- If the PROCHOT_POL bit is set low, when PROCHOT# goes low, the PROCHOT_STS bit will be set. This indicates that the thermal threshold has been exceeded. If the PROCHOT_EN bit is set, then when PROCHOT_STS goes active, either an SMI# or SCI# will be generated (depending on the SCI_EN bit being set). The power management software (pre-boot firmware or ACPI) can then take measures to start reducing the temperature. Examples include shutting unneeded subsystems or halting the processor. Another method to reduce the temperature is processor-

initiated passive cooling. Refer to Section 50.2.2.2, “Processor Passive Cooling” for more information.

- By setting the PROCHOT_POL bit to high, another SMI# or SCI# can optionally be generated when the PROCHOT# signal goes back high. This allows the software (pre-boot firmware or ACPI) to turn off the cooling methods.
- PROCHOT# assertion does not cause a TCO event message in S3 or S4. The level of the signal is not reported in the heartbeat message.

50.2.2.2 Processor Passive Cooling

The EP80579 provides a processor-initiated passive cooling feature that actives a clock throttling mechanism, which can be used by power management software or pre-boot firmware to reduce the temperature.

The FORCE_THTL bit allows the pre-boot firmware to force passive cooling independent of the ACPI software (which uses the THTL_EN and THTL_DTY bits). The FORCE_THTL bit has the following behavior:

- If the FORCE_THTL bit is set, the EP80579 will start throttling using the ratio in the PROCHOT_DTY field. The duty cycle indicates the approximate percentage of time the STPCLK# signal is asserted while in thermal throttle mode. The STPCLK# throttle period is 1024 PCICLKs (PCICLK = 33 MHz). Throttling only occurs if the system is in the C0 state. If in the C2, C3, or C4 state, no throttling occurs.

  Note: Refer to Table 50-2 for duty cycle ratio settings.
- If the FORCE_THTL bit is turned off (cleared), the EP80579 will stop throttling, unless the THTL_EN bit is set (indicating that ACPI software is attempting throttling).
- If both the THTL_EN and FORCE_THTL bits are set, then the IICH uses the duty cycle defined by the PROCHOT_DTY field, not the THTL_DTY field. (PROCHOT_DTY has higher priority).
- Once the PROCHOT_DTY field is written, subsequent writes have no effect until PLTRST# goes active.

50.2.2.3 On-Demand Passive Cooling

The EP80579 provides an on-demand passive cooling feature, which may be used in customer designs that require power management capabilities that take advantage of clock throttling. The on-demand feature is similar to the processor passive cooling in that it supports clock throttling, but the on-demand feature is initiated by software setting the THTL_EN and THTL_DTY bits.

The on-demand passive cooling feature behaves as follows:

- Software sets the THTL_DTY bits to select throttle ratio and the THTL_EN bit to enable the throttling.
- Throttling results in STOPCLK# active for a minimum time of 12.5% and a maximum of 87.5%. The period is 1024 PCI clocks (PCICLK = 33 MHz). Thus, the STOPCLK# signal can be active for as little as 128 PCI clocks or as much as 896 PCI clocks. The actual slowdown (and cooling) of the processor will depend on the instruction stream, because the processor is allowed to finish the current instruction. Furthermore, the EP80579 waits for the STOP-GRANT cycle before starting the count for the time the STOPCLK# signal is active.

  Note: Refer to Table 50-2 for duty cycle ratio settings.

Table 50-2. PROCHOT_DTY/THTL_DTY Throttle Ratios

Throttle Mode	PCI Clocks
Default (will be 50%)	512
87.5%	896
75.0%	768
62.5%	640
50%	512
37.5%	384
25%	256
12.5%	128

§ §